Summer 8-21-2016

An Application of Wavelet Methods to the Search for Baryon Acoustic Oscillations

Kevin Michael Kadowaki
DePaul University, kkadowaki@luc.edu

AN APPLICATION OF WAVELET METHODS TO THE SEARCH FOR BARYON ACOUSTIC OSCILLATIONS

A Thesis

Presented in

Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

August, 2 0 1 6

BY

Keven Kadowaki

PHYSICS DEPARTMENT

College of Science and Health

DePaul University

Chicago, Illinois

TABLE OF CONTENTS

LIST OF FIGURES **5**

ABSTRACT **7**

0.1 Introduction 8

0.2 Acknowledgements 9

CHAPTER 1 Large Scale Structure **11**

1.1 Baryonic Acoustic Oscillations 11

1.1.1 The Early Universe 11

1.1.2 The Significance of BAOs 14

1.2 Clustering Statistics 15

1.2.1 2-Point Correlation Function 16

1.2.2 Power Spectrum 17

1.3 SDSS and Previous Results 19

1.3.1 Overview of SDSS 19

1.3.2 Previous Measurement and Consensus Values 21

1.3.3 Our Use of SDSS Resources 24

CHAPTER 2 Wavelet Theory **26**

2.1 Wavelet Basics 26

2.1.1 The Haar Wavelet 27

2.1.2 The Daubechies-2 Wavelet 28

2.2 Wavelet Transforms 30

2.2.1 Signal Approximation 31

2.2.2 The Discrete Wavelet Transform 35

TABLE OF CONTENTS – *Continued*

2.2.3 The Wavelet Packet Transform 43
2.3 2-D Transforms 46
2.4 Wavelet vs. Fourier Methods 47

CHAPTER 3 Spectrum Estimation 49
3.1 Procedure Overview 49
3.2 Preparing the Data 50
3.2.1 SDSS Instrumentation 50
3.2.2 Celestial Coordinates 51
3.2.3 Weighting 52
3.2.4 HealPix and SDSSPix 54
3.3 Producing the Power Spectrum 57
3.3.1 Parseval's Theorem 57
3.3.2 Spectrum Estimation 58
3.4 Radial Averaging 63

CHAPTER 4 Data Analysis 66
4.1 Power Spectrum Analysis 66
4.1.1 Extracting the BAO Signal 66
4.1.2 Computing α 67
4.1.3 Computing β 68
4.2 Code 69
4.2.1 Overview and `main` 69
4.2.2 Subroutines 71
4.3 Results 72

CHAPTER 5 Conclusions and Future Research 78
5.1 Interpretation of Results 78

TABLE OF CONTENTS – *Continued*

5.2 Future Research 81

Appendices 83

APPENDIX A Proving Parseval's Theorem for Wavelet Transforms 84

APPENDIX B `sdsspix` Program 87

APPENDIX C `matrice` Program 90

APPENDIX D `WPS` Program 92

APPENDIX E `alphafitter` Program 95

LIST OF FIGURES

1.1 Evolution of the BAO from a spherically symmetric perturbation. . . 12
1.2 Sample correlation function with BAO bump. 14
1.3 An example of a power spectrum/correlation function pair. 19
1.4 Boss survey footprint 21
1.5 Power spectrum, smooth and oscillatory components 23

2.1 The Haar scaling function and wavelet 28
2.2 The Daubechies-2 scaling function and wavelet 30
2.3 Example of a signal and its Haar approximation 32
2.4 Example of a signal and its Daubechies approximation 35
2.5 Tree representation of DWT 38
2.6 Example of the Haar DWT, First Pass 38
2.7 Example of the Haar DWT, Second Pass 40
2.8 Tree representation of WPT 44
2.9 Tree representation of example WPT 45
2.10 Tree representation of the 2 dimensional WPT 47

3.1 Procedural flowchart 50
3.2 A depiction of SDSS coverage 51
3.3 HEALPix indexing schemes 55
3.4 SDSSPix Result 56
3.5 Wavelet Transform as Action of High- and Low-Pass Filters 59
3.6 Natural vs. Frequency Ordering 61
3.7 Frequency Grid and Greycode Matrix 62

LIST OF FIGURES – *Continued*

3.8 Sample Two-Dimensional Power Spectrum 64

3.9 Sample Radially Averaged Power Spectrum 65

4.1 Oscillatory components of the power spectrum and fiduciary power spectrum 68

4.2 Procedural flowchart 70

4.3 α and β values, Northern dataset, 'ppd' 74

4.4 α and β values, Southern dataset, 'ppd' 74

4.5 α and β values, Northern dataset, 'sym' 75

4.6 α and β values, Southern dataset, 'sym' 75

4.7 Averages and Spreads of α 76

4.8 Averages and Spreads of β 77

ABSTRACT

We use the method of wavelet transforms to investigate the clustering of matter on galactic scales in search of Baryon Acoustic Oscillations (BAOs). In particular, we develop a method of wavelet packet analysis to measure the power spectrum of a galaxy distribution and apply this method to the CMASS galaxy catalogue from the Sloan Digital Sky Survey (SDSS) Baryon Oscillation Spectroscopic Survey (BOSS) collaboration. Using a variant of HEALPix, we project the data from the celestial sphere onto a square signal matrix, and take the two-dimensional Wavelet Packet Transform of this signal. We use this transform and techniques of spectrum estimation to estimate the power spectrum, and extract the BAO signature from this measured power spectrum. Using fitting methods, we compare our results to a fiducial ΛCDM flat cosmological model and previous results from the BOSS collaboration, and we detect a BAO signature in the power spectrum comparable to the previous consensus results of the BOSS collaboration. In reproducing these results, our method of spectrum estimation is an improvement on established methods insofar as it requires no free nuisance parameters, other secondary manipulations, or mock catalogues. In particular, we calculate two parameters that determine how well our results fit previous studies and models, where a value of 1 for a given parameter indicates a perfect fit. For the parameter α, which measures how well our data fits a fiduciary concordance ΛCDM model, we find $\alpha \approx .94$. For the parameter β, which measures how well our BAO matches the results from the SDSS collaboration, we find that $\beta \approx 1$. In both parameters there are slight variations depending on the particular wavelet used and the resolution of the signal.

0.1 Introduction

We begin—as all astrophysics must, implicitly or explicitly—with the Cosmological Principle: on sufficiently large scales, the universe is roughly homogenous and isotropic. Given that we notice some pretty obvious inhomogeneities on the human, stellar, and galactic scales, this immediately leads us to ask what causes the universe to depart from homogeneity at scales below a certain margin. As it turns out, knowledge of how these inhomogeneities arise and their precise nature gives us significant tools for understanding the structure and evolution of the universe.

The purpose of this thesis is to use the methods of wavelet transforms to investigate the clustering of matter on galactic scales in search of Baryonic Acoustic Oscillations (henceforth, BAOs). These BAOs are an aftereffect of the Big Bang, and provide a 'standard ruler' for measuring the crucial distance-redshift relation [1].

The section on Large-Scale Structure (Chapter 1) lays out the basic statistical tools for our investigation—in particular, the power spectrum—and explains how BAOs arose from the coupling of baryons and photons in the early universe. This chapter will also review some of the literature on the Sloan Digital Sky Survey (SDSS) and present the previous consensus values for the measurement of BAOs. Chapter 2, outlines the theory behind the discrete wavelet transform (DWT) and the wavelet packet transform (WPT). Chapter 3 describes the methods by which wavelet transforms can be used in spectrum estimation. In Chapter 4, we will present the technical details and the code used, as well as the results of our analysis. Finally, in Chapter 5, we will interpret our results, draw our conclusions, and discuss implications for future research.

0.2 Acknowledgements

First and foremost, I would like to thank my advisor, Dr. Jesus Pando—when I began this project, I knew very little about astrophysical research and nothing about large-scale structure. But he did not merely teach me the tools of the trade. I have also learned certain attitudes and ways of seeing that one takes up in the process of physics research, and I feel this, albeit difficult to describe, was by far the most important and fundamental part of my education.

I would also like to thank my parents, who have given me emotional, spiritual, and financial support to allow me to undertake my studies at DePaul. My ongoing pursuit of an academic career has been and will continue to be a long and difficult journey, and I am grateful for all the support they have given me along the way.

I would like to thank the rest of the faculty at DePaul's physics department, for the positive learning environment and knowledge that they have bequeathed to me. In particular, I would like to thank Dr. Anuj Sarma and Dr. Bernhard Beck-Winchatz for their service on my thesis committee. I would like to thank the undergraduate research assistants who worked with me on this project, Noel Garcia and Taurean Ford, as well as all of my peers at DePaul who have helped me along the way.

Finally, this research would not have been possible without the financial and scientific support of the Sloan Digital Sky Survey IV, and the FAST program that funded our research. In particular, I owe tremendous thanks to our FAST team coordinator, Kelly Holley-Bockelmann, and our FAST team collaborators from Carnagie Mellon, Shirley Ho and Anthony Pullen.

Funding for the Sloan Digital Sky Survey IV has been provided by the Alfred P. Sloan Foundation, the U.S. Department of Energy Office of Science, and the Participating Institutions. SDSS-IV acknowledges support and resources from the Center for High-Performance Computing at the University of Utah. The SDSS web site is

www.sdss.org.

SDSS-IV is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS Collaboration including the Brazilian Participation Group, the Carnegie Institution for Science, Carnegie Mellon University, the Chilean Participation Group, the French Participation Group, Harvard-Smithsonian Center for Astrophysics, Instituto de Astrofsica de Canarias, The Johns Hopkins University, Kavli Institute for the Physics and Mathematics of the Universe (IPMU) / University of Tokyo, Lawrence Berkeley National Laboratory, Leibniz Institut fr Astrophysik Potsdam (AIP), Max-Planck-Institut fr Astronomie (MPIA Heidelberg), Max-Planck-Institut fr Astrophysik (MPA Garching), Max-Planck-Institut fr Extraterrestrische Physik (MPE), National Astronomical Observatory of China, New Mexico State University, New York University, University of Notre Dame, Observatrio Nacional / MCTI, The Ohio State University, Pennsylvania State University, Shanghai Astronomical Observatory, United Kingdom Participation Group, Universidad Nacional Autnoma de Mxico, University of Arizona, University of Colorado Boulder, University of Oxford, University of Portsmouth, University of Utah, University of Virginia, University of Washington, University of Wisconsin, Vanderbilt University, and Yale University.

CHAPTER 1

Large Scale Structure

In this chapter, we sketch a brief overview of the background for this project. We begin by outlining the cosmology behind BAOs—how they arise, and why there are significant. We will then cover the primary statistic that we are trying to measure, the power spectrum, by means of understanding its Fourier transform, the 2-point correlation function. However, we will not discuss our methods of spectrum estimation here. We will save that discussion for Chapter 3. Finally, we will give a brief overview of the Sloan Digital Sky Survey (SDSS) and review their consensus values for BAOs.

1.1 Baryonic Acoustic Oscillations

1.1.1 The Early Universe

Shortly after the Big Bang, the universe consisted of a dense, opaque plasma soup of tightly coupled baryons and photons, on a background of largely homogeneous dark matter. Slight inhomogeneities in the dark matter density field eventually led to dark matter gravitational wells, into which baryonic matter collapsed. But because this plasma was so energetic, and because radiation was tightly coupled to the baryons by Thompson scattering, the total collapse of the universe was prevented by radiation pressure. Instead, this contest between gravity and radiation pressure caused these initially small perturbations to send pressure waves throughout the plasma soup of the universe.

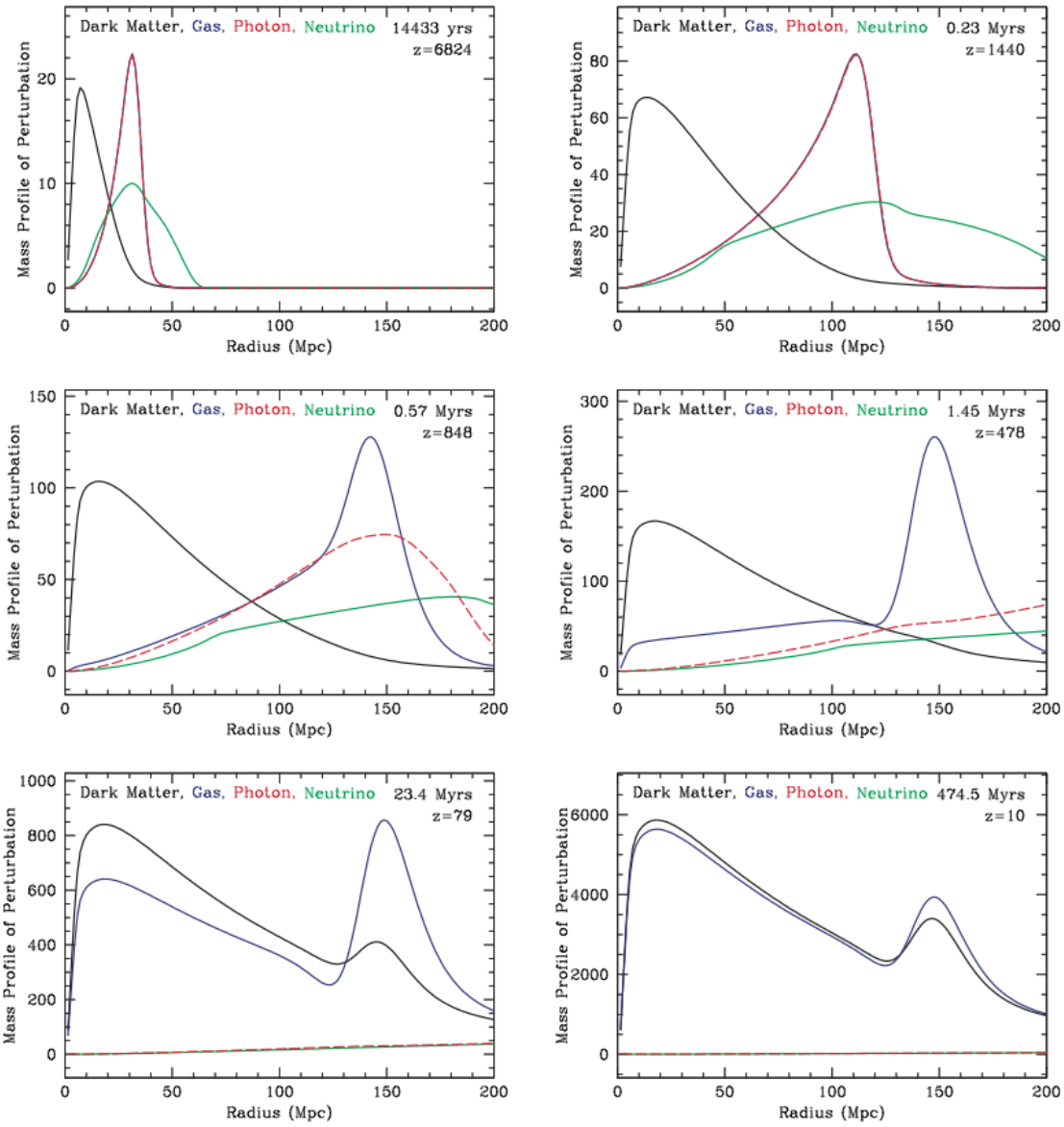

Figure 1.1: The propogation of a coupled baryon-photon wave caused by a spherically symmetric perturbation in the density field. The units of the mass profile are arbitrary, but are scaled between the various plots. See text for an extended description of the transition between the various plots.

Figure 1.1 (from [2]) is a simplified picture of this process which shows us the progression of one such spherically symmetric perturbation in an otherwise completely

homogeneous universe. After an initial slight relative overdensity of matter (top left), the coupled baryons and photons propagate together outwards in a pressure wave (top right). As the universe expands and the energy density lowers, the formerly ionized electrons recombine with baryons, and this leads to the decoupling of baryons and photons (middle left). As these photons stream away, the matter wave of baryons—no longer subject to radiation pressure—settles (middle right). Finally, the gravitational interaction between baryons and dark matter causes them to coalesce, but not without leaving the imprint of the original baryon wave (bottom right and left) [3]. These 'frozen-in waves' then leave their marks as a characteristic scale in the subsequent formation of stars and galaxies. Because they resulted from pressure waves (like sound waves), they are known as Baryon Acoustic Oscillations, or BAOs.

Crucially, the characteristics of the matter wave that imprints the BAO depends on the speed of the wave and the amount of time before recombination, which in turn depend on the ratios of matter to radiation ($\Omega_m h^2$) and baryons to photons ($\Omega_b h^2$) [4]. Thus, the BAO signal depends on the cosmology of the universe, and by measuring the BAO we can extract important cosmological information.

Of course, the example in Figure 1.1 is extremely simple. In the real universe, there was not merely one perturbation. Rather, there were many perturbations occurring simultaneously all over the universe. Thus, while it is easy identify the BAO in Figure 1.1, real-world measurements of the BAO signal require more powerful statistical tools. Nevertheless, if the picture described above is correct, the BAO will be visible in the two-point correlation function (described in detail in Section 1.2.1) as an extra bump around 100-150 Mpc h^{-1}. Figure 1.2 is an example of just such a two-point correlation function from [1], and in particular the peak that represents the BAO signal can be seen at approximately 110 Mpc h^{-1}. The precise placement of the peak is what we are trying to measure.

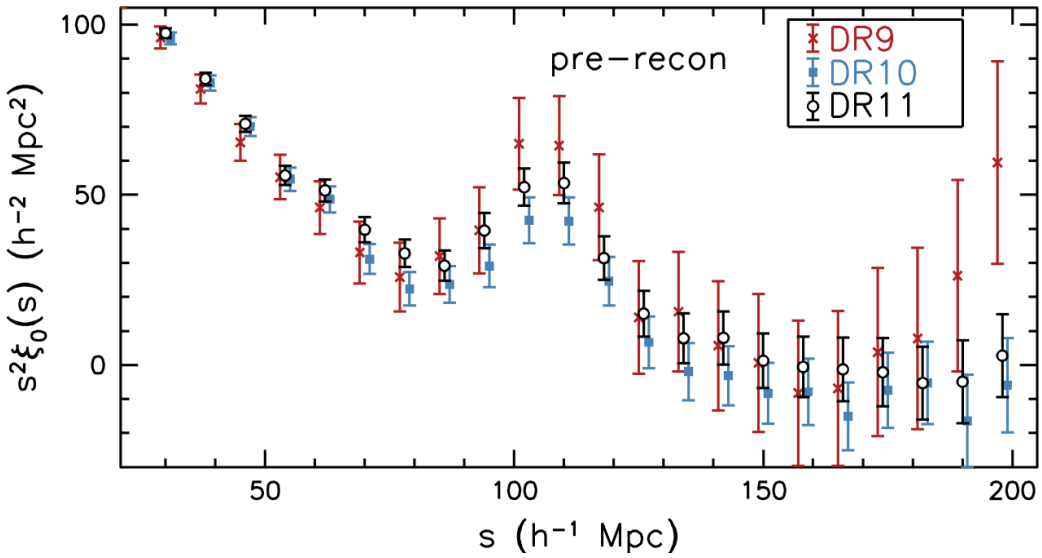

Figure 1.2: Figure from [1]. The x-axis represents the scale s, and the y-axis shows the likelihood of excess clustering at that scale (convolved with s^2 to make the bump more prominent). Note the bump in the correlation function at around 110-120 Mpc h^{-1}—this is the BAO signature. This bump is the distinctive feature, as a universe that had never undergone the sort of perturbations outlined above in Fig. 1 would lack this bump. In our research, we attempt to measure the placement of this bump by measuring the Fourier transform of this correlation function.

1.1.2 The Significance of BAOs

From a purely theoretical perspective, because the physics of the BAO relies on dark matter seeding the initial gravitational wells, the existance of the BAO signal confirms the existance of dark matter. Thus, the measurement of BAO is strong evidence against certain cosmological models that attempt to dispense with dark matter, such as Modified Newtonian Dynamics (MOND).

From a more practical perspective, precisely because the BAO placement depends on cosmological parameters, a measurement of the BAO signal can help us constrain these parameters. BAO measurements, alongside other key parameter-constraining measurements (such as Type Ia supernovae and the CMB), are therefore crucial for

laying the foundations of cosmology generally.

In particular, BAOs serve as a 'standard ruler' for measuring cosmological distances. Since the BAO length scale is the same everywhere in the universe, we can use information from the the BAO signature to fix the distance-redshift relationship. For example, in the radial power spectrum we can measure the Hubble parameter, and from the angular power spectrum we can measure the angular diameter distance [3]. Combined with other methods of probing cosmological parameters, such as CMB anisotropy and Type Ia supernovae, measurement of the BAO signal has allowed astrophysicists to test models of dark energy and further substantiate the model of a flat (i.e., no intrinsic curvature) universe [5].

1.2 Clustering Statistics

In cosmology, observations have shown that matter is not randomly distributed. The study of large scale structure is is the study of how matter is *clustered* differently than in a random distribution. We can quantify this as follows.

If $\rho(\vec{r})$ is the number density of objects at $\vec{r}$, and $\bar{\rho} = \langle \rho(\vec{r}) \rangle$ is the mean number density over the field, it is convenient to define the dimensionless overdensity $\delta(\vec{r})$ as

$$\delta(\vec{r}) \equiv \frac{\rho(\vec{r}) - \bar{\rho}}{\bar{\rho}} \tag{1.1}$$

The quantity $\delta(\vec{r})$ measures the difference of the density at a given point from the average density. The fluctuations given by $\delta(\vec{r})$ change under the influence of gravity and the way these cluster (or don't) will provide the primary way of distinguishing the distribution of matter from a random distribution. The behavior of this quantity is what we will use to quantify clustering.

1.2.1 2-Point Correlation Function

The measure that is used to characterize clustering in large scale structure study is the *two-point correlation function*. The quantity is essentially a probability measure that gives the probability of finding an second object in a specified annular ring about a first object. It is given by [6]

$$\xi(\vec{r}) = \langle \delta(\vec{x} + \vec{r})\delta(\vec{x}) \rangle. \tag{1.2}$$

which is simply the correlation of the overdensity function with an overdensity function shifted by $\vec{r}$. Because we assume homogeneity, we know that our function is translationally invariant; thus, our function depends only on the vector separating the two vectors $\vec{r}$. Similarly, based on our assumption of isotropy, our function is rotationally invariant; thus, it is a function merely of the distance relation between the two vectors [7]. Thus we can write that

$$\xi(\vec{r}) = \xi(r) = \langle \delta(\vec{x} + \vec{r})\delta(\vec{x}) \rangle. \tag{1.3}$$

Using the definition of the overdensity Equation 1.1, we can show that Equation 1.3 can be written as

$$\bar{\rho}^2[1 + \xi(r)] = \langle \rho(\vec{x} + \vec{r})\rho(\vec{x}) \rangle \tag{1.4}$$

This formulation makes clear what the correlation function actually measures. Given two small regions dV_1 and dV_2 picked out by $\vec{r}_1$ and $\vec{r}_2$, the expected number of pairs of galaxies, n_p, found with one in each volume element is

$$n_p = \bar{\rho}^2[1 + \xi(r)]dV_1 dV_2 \tag{1.5}$$

In a truly random distribution, the probability of finding an object in dV_2 would be independent of the probability of finding an object in dV_1, and both probabilities would be equal to $\bar{\rho}dV$. Basic statistics tells us that the joint probability of two

statistically independent events is just the product of their respective probabilities; therefore, the expected number of pairs of galaxies in a random distribution is

$$n_{p,rand} = \bar{\rho}^2 dV_1 dV_2, \tag{1.6}$$

or that

$$n_p = n_{p,rand} + \bar{\rho}^2 \xi(r) dV_1 dV_2 \tag{1.7}$$

Thus, we see that the correlation function measures the *deviation* from this random distribution, or fraction of galaxy pairs in an annular ring a distance r above (or, if negative, below) the amount we would expect from a random distribution [6]. Intuitively, if we imagine throwing down a bundle of R-length sticks onto a distribution of objects, the correlation function $\xi(R)$ measures the ratio of the number of sticks that statistically fall with objects at both ends compared to the result we would get if we performed this operation on a distribution of objects that is totally random [8].

If we have a spike in the correlation function at r_s, this simply means that there is a higher probability of us finding objects separated by this distance r_s than other distances. If ξ has a spike of this sort, we call r_s the 'characteristic scale' of the distribution.

1.2.2 Power Spectrum

The two-point correlation function has a clear physical meaning and is therefore the starting-off point for talking about clustering. However, the two-point correlation function is computationally expensive. It is also greatly affected by Poisson noise, and the distances at which correlations measured are in some sense arbitrary. Many of these issues are alleviated by using the correlation function's Fourier transform, the power spectrum. Because numerically stable and fast algorithms exist for computing Fourier transforms, in many large scale structure studies it is the power

spectrum that one actually tries to numerically compute. Once this is computed, one can take advantage of the invertability of the Fourier transform to obtain the two-point correlation function.

In our work, we will be using two-dimensional data spread along the celestial sphere. By the Wiener-Khinchin theorem, the power spectrum and the two-point correlation function are related by the Fourier transform. And because they are spherically symmetric, the Fourier transform can be written as a Hankel transform, with the transform pairs given by

$$P(k) = \int_0^\infty \xi(r) J_0(kr) r \, dr \tag{1.8}$$

$$\xi(r) = \int_0^\infty P(k) J_0(kr) k \frac{dk}{2\pi} \tag{1.9}$$

where $P(k)$ is the power spectrum, J_o is a Bessel function of the first kind.

We can get an intuitive understanding of what the power spectrum measures by using an explicit example. Imagine that the power spectrum consists of a delta function, $P(k) = \delta_D(k - k_0)$. It is a simple matter to solve for the two-point correlation function

$$\xi(r) = \frac{k_0}{2\pi} J_0(k_0 r) \tag{1.10}$$

Thus, the two-point correlation function corresponding to $P(k) = \delta_d(k - k_0)$ is just a 0th Bessel function of the first kind scaled by k_0 along both axes and a factor of 2π (see Figure 3).

We can understand the power spectrum in a similar way that we understood the correlation function—except instead of throwing down sticks, we would throw down Fourier 'waves' [8]. In our example, where objects are all found at distances which are an integer multiple of a fixed distance, this means that Fourier waves of wave number k_0 are better representative of the density field than they would be for a

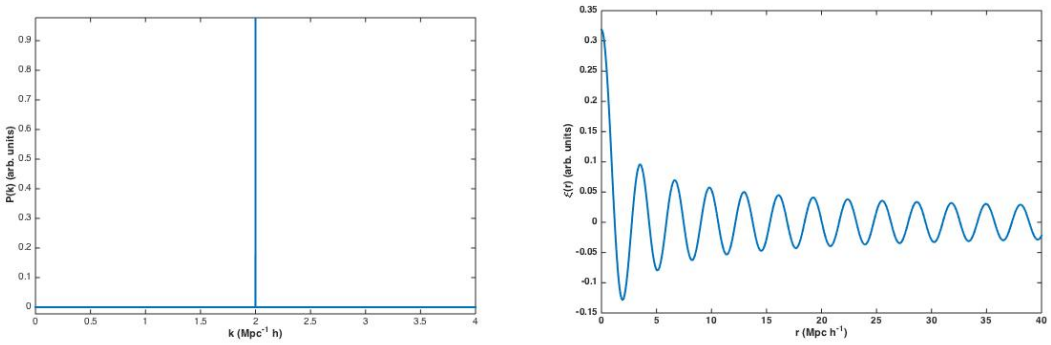

Figure 1.3: Left: (a) a sample power spectrum, which is just a delta function centered at $k = 2$. Right: (b) the two-point correlation function corresponding to the delta function power spectrum. Note how there are spikes in the correlation function at approximately even intervals along the r-axis.

random distribution of galaxies. Just as the correlation function gives us the variance as a function of scale in real space, the power spectrum tells us the variance as a function of scale in frequency space.

1.3 SDSS and Previous Results

In order to study large scale structure, a large and fairly complete data set is necessary to overcome the noise associated with these measurements. Fortunately, observational astronomy has been extremely fruitful in the last few decades, and data sets that meet these requirements have indeed been produced. In this thesis, we will be using data from the Sloan Digital Sky Survey.

1.3.1 Overview of SDSS

The Sloan Digital Sky Survey (SDSS) is a wide-field imaging and spectroscopic survey which has been running since April 2000. It consists of a number of sub-collaborations aimed at projects on subjects as wide-ranging as galaxy evolu-

tion (APOGEE), extra-solar planet discovery (MARVELS), and galactic structure (SEGUE-2). The project uses a 2.5 meter telescope based in Apache Point, NM, USA for spectroscopic analysis of target-selected objects in the night sky covering approximately 1/3 of the celestial sphere. Through a series of 12 cumulative data releases from 2000-2014, survey data from the three major installments of the SDSS survey (identified by SDSS-I, SDSS-II, and SDSS-III) has been made available to the public for scientific research. These data releases include both the raw spectroscopic and imaging data, as well as processed data and object catalogues [9]. As of this writing, SDSS is currently undergoing its fourth installment, SDSS-IV, which will begins operations on an additional 2.5 meter telescope at Las Campanas Observatory in Chile as well as significantly expand the scope of the previous installments.

The particular sub-collaboration that concerns this thesis is the Baryon Oscillation Spectroscopic Survey (BOSS), and the extension of this project for SDSS-IV, eBOSS. Objects targeted for BOSS/eBOSS were selected specifically for redshifts $z < .7$ in two continuous regions of sky covering approximately 10400 deg^2 (see Figure 1.4, from [9]), resulting in a total of approximately 2.5 million spectra [9]. Once spectra were obtained from these targets, the spectra were processed by the BOSS data pipeline to determine redshift and object classification (star, galaxy, quasar, etc.); details regarding this process are described in [10]. Classified objects can then be grouped into various catalogues which correspond to the various uses researchers will make of the data.

As of Data Release 12, this BOSS data has already been used to advance our understanding of cosmology on numerous fronts—among other things, the BAO signal has been measured in a number of galaxy clustering studies (summary analysis in [1]), the neutrino mass upper bound measurement has been improved [11], and constraints on various cosmological parameters have been improved [5]. A more complete listing of the research associated with both the BOSS collaboration and

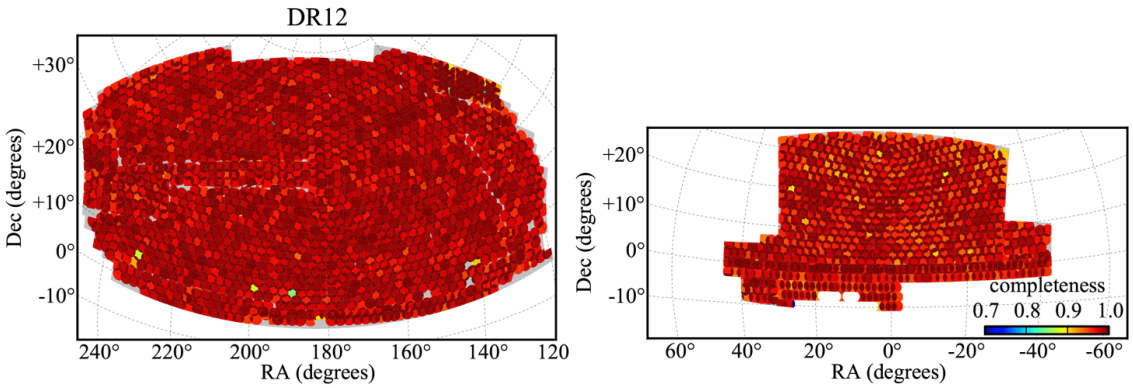

Figure 1.4: Footprint for BOSS survey as of DR12 in the (a) Northern galactic cap (left), and the (b) Southern galactic cap (right). The degrees of redness indicate the level of completeness of the survey. Figure from [9]

the SDSS-III collaboration as a whole can be found in [9]. Of particular significance for us is the work summarized by [1] and [12]; these papers, which give a consensus values for the BAO signal in galaxy clustering studies done by the collaboration through DR10/11 and DR12, respectively, have helped to guide our research in a number of respects (see Section 1.3.3).

1.3.2 Previous Measurement and Consensus Values

The most recent measurements of the BAO signal from the SDSS collaboration have been derived from DR12, in [12]. Using the Constant Stellar Mass (CMASS, redshift $0.43 < z < .70$) and low-redshift sample (LOWZ, redshift $0.15 < z < 0.43$) catalogues and the Landy-Szalay estimator [13], the correlation function is estimated as a series of datapoints [12].

The placement of the BAO signal is reported in terms of parameter α, which is defined as the ratio between the measured value and a fiduciary cosmology in terms of the angular diameter distance $D_A(z)$, the Hubble parameter $H(z)$, and the sound horizon r_d. The parameters of the fiduciary cosmology itself, given in [14], are

determined by the collaboration itself and are in general similar to the best-fit WMAP 7 year model.

$$\alpha = \frac{D_A^{2/3}(z)H^{-1/3}(z)/r_d}{(D_A^{2/3}(z)H^{-1/3}(z)/r_d)_{fid}} \tag{1.11}$$

Using a fiducial cosmological model, an expected correlation function $\xi_{fid}(r)$ is generated. The datapoints of the measured correlation function are then fit to a function involving said fiducial function; the actual placement of the BAO signal is reported in terms of how much the fiducial model must be stretched (or contracted) to fit the actual datapoints. In particular, the following model has been used,

$$\xi^{fit}(r) = B_0^2\xi_{fid}(\alpha r) + A_0 + \frac{A_1}{r} + \frac{A_2}{r^2} \tag{1.12}$$

where A_0, A_1, and A_2 are extra coefficients to account for possible biases, and B_0 is a normalization factor. Since the α parameter stretches or compresses the fiduciary model along the r-axis, it adjusts the placement of the BAO bump to match the location of the BAO bump in the measured two-point correlation function. In this way, α reports the location of the BAO signal as a measure of the deviation from the fiduciary cosmology. The result from CMASS and LOWZ catalogue are $\alpha = 1.0051 \pm 0.0098$ and $\alpha = 1.026 \pm 0.024$, respectively.

The parameter α can also be calculated from the measurement of the power spectrum, using a similar methodology. Using the Feldmen-Kaiser-Peacock estimator [15], the power spectrum $P^{fit}(k)$ is calculated and once more compared with the power spectrum of a fiducial cosmology produced by the Code for Anisotropies in the Microwave Background (CAMB) [16] by means of a fit scaled by the parameter α.

In the same way that the BAO signal shows up as a 'bump' in the correlation function, the BAO signal manifests itself in the power spectrum as an oscillatory component convolved with the background power spectrum. Thus, to perform a proper fit between the fiduciary model and the measured power spectrum, the fidu-

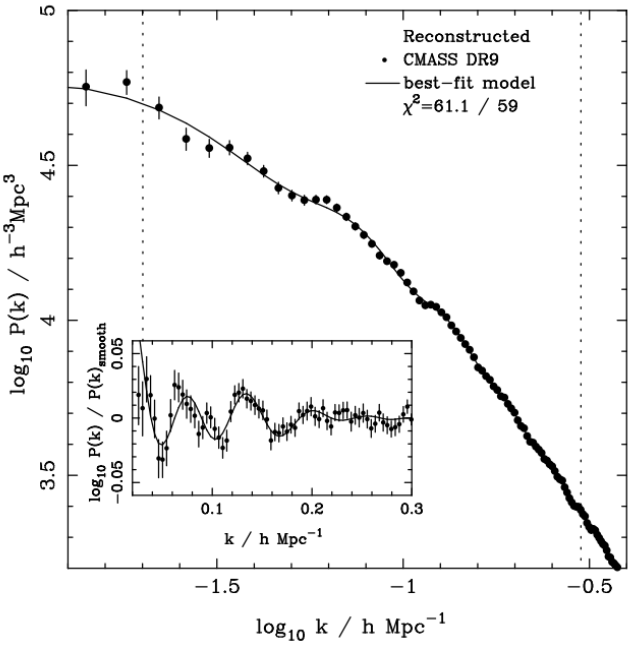

Figure 1.5: From [14]. The power spectrum can be separated into two components. The smooth component $P^{sm}(k)$ is represented by the best fit line, and the oscillatory component is equal to $O^{lin}(k) = P(k)/P^{sm}(k)$. Note that while the graph of the full power spectrum is a log-log graph, the inset oscillatory component of the power spectrum is merely a log graph.

ciary power spectrum must be separated into smooth and oscillatory components, as illustrated in Figure 1.5. In this figure, the line represents the smooth component of the power spectrum P^{sm}, while the data points that oscillate around this function are the result of the BAO. The inset graph shows us the oscillatory component of the power spectrum, $O^{lin}(k) = P(k)/P^{sm}(k)$.

Once the fiduciary cosmology has been divided into its $P^{sm}(k)$ and $O^{lin}(k)$ components, we can fit the measured power spectrum according to the formula

$$P^{fit}(k) = P^{sm}(k)\left[1 + (O^{lin}(k/\alpha) - 1)e^{-\frac{1}{2}k^2\Sigma_{nl}^2}\right] \tag{1.13}$$

where P^{fit} is the measured power spectrum, P^{sm} and O^{lin} are smooth and oscillatory components of the fiducial cosmology generated by CAMB, and Σ_{nl} is a parameter

that depends on the galaxy sample [1]. In a similar way the α parameter adjusts the position of the BAO signal in the fiduciary two-point correlation function, the α parameter in the power spectrum adjusts the precise frequency of the oscillatory component of the fiduciary power spectrum to find the best fit to the oscillatory component of the measured power spectrum.

The fitting methods used by SDSS for the two-point correlation function and the power spectrum both involve the use of somewhat arbitrary nuisance parameters. While we still wish to calculate α, one of our goals is to develop a method of measuring alpha that does not rely on all these extra fitting parameters, in hopes that our measurement will be more faithful to the data. This process of calculating α will be covered in greater detail in Chapter 3, as we will need to use it to report our own results for comparison with SDSS consensus values.

1.3.3 Our Use of SDSS Resources

Our collaboration with SDSS will assist our research in two major respects. First, we will use the consensus values obtained by the SDSS team to ensure that our own methods are accurate—that is, we should be able to run our algorithms on the same datasets analyzed in [1; 12] and obtain similar (though not necessarily identical) results. Thus, the finished work done by the SDSS collaboration will serve as a litmus test for us to check that our own methods are working correctly. As well, we can use SDSS's consensus values and error bars to minimize the computational workload in our calculations of α—this procedure will be described in more detail in Chapter 3.

Second, our collaboration with SDSS gives us access to the raw data of SDSS-IV pre-public release. With this data, we can use our methods—honed on the previous SDSS-III data releases and checked against other collaboration results—to obtain a

better, more constrained value for α.

CHAPTER 2

Wavelet Theory

The main mathematical tools that we will use in our investigation are called *wavelets*. In this chapter, we will introduce the theory that underlies wavelet methods. We will begin by outlining the basics of what wavelets are and how they work (Section 2.1). Next, we will describe the process of the 1-dimensional discrete wavelet transform (DWT) and wavelet packet transform (WPT) (Section 2.2). Third, we will extend the transforms of Section 2.2 to the 2-dimensional case, as we will need to do these transforms on 2-dimensional galaxy datasets (Section 2.3). Finally, we will outline the general advantages and disadvantages of wavelet methods in comparison to the more well-known Fourier methods (Section 2.4), and explain why we believe wavelet methods are particularly suitable to our specific project. Unless otherwise stated, all information presented in this chapter is drawn from [17].

2.1 Wavelet Basics

"Wavelets" are the name of a *class* of special functions which satisfy certain requirements. Like Fourier waves, we need them to have some kind of oscillatory behavior, and we generally want them to comprise an orthonormal basis. Unlike Fourier waves, however, these functions integrate to zero over infinity and are well localized [18]. A full discussion follows in Section 2.2. While there are many functions that can serve as wavelets, to get a sufficient understanding of wavelets, we can limit our treatment of wavelets to a specific subset called the *Daubechies* wavelets. These are also the class of wavelets used in this thesis work.

2.1.1 The Haar Wavelet

The Haar wavelet is the simplest of the Daubechies wavelets. As such, it will serve us well as an introduction to what wavelets are. We begin with a simple unit step function

$$\phi(x) = \begin{cases} 1 & \text{if } 0 \leq x < 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

This function is simply equal to 1 over the interval $[0, 1)$, and zero everywhere else. See Figure 2.1a for a visual of this step function. Notice that this simple unit step function can then easily be stretched and translated by

$$\phi_{[a,b)}(x) = \phi\left(\frac{x-a}{b-a}\right) = \begin{cases} 1 & \text{if } a \leq x < b \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

so that we can obtain a step function along any interval we like. In Eq. (2.2) the interval is $[a, b)$ (See Figure 2.1a for a visual of the scaling function ϕ over the interval $[0, 1)$). Using this definition of the Haar scaling function, we can define the Haar wavelet

$$\psi_{[0,1)}(x) = \phi_{[0,\frac{1}{2})}(x) - \phi_{[\frac{1}{2},1)}(x) = \begin{cases} 1 & \text{if } 0 \leq x < \frac{1}{2} \\ -1 & \text{if } \frac{1}{2} \leq x < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

Similarly, over a generic interval we have,

$$\psi_{[a,b)}(x) = \phi_{[a,\frac{a+b}{2})}(x) - \phi_{[\frac{a+b}{2},b)}(x) = \begin{cases} 1 & \text{if } a \leq x < \frac{a+b}{2} \\ -1 & \text{if } \frac{a+b}{2} \leq x < b \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

In both of these cases, the wavelet breaks the interval in question in half and takes a constant value of 1 in the first half, and -1 in the second half. See Figure 2.1b

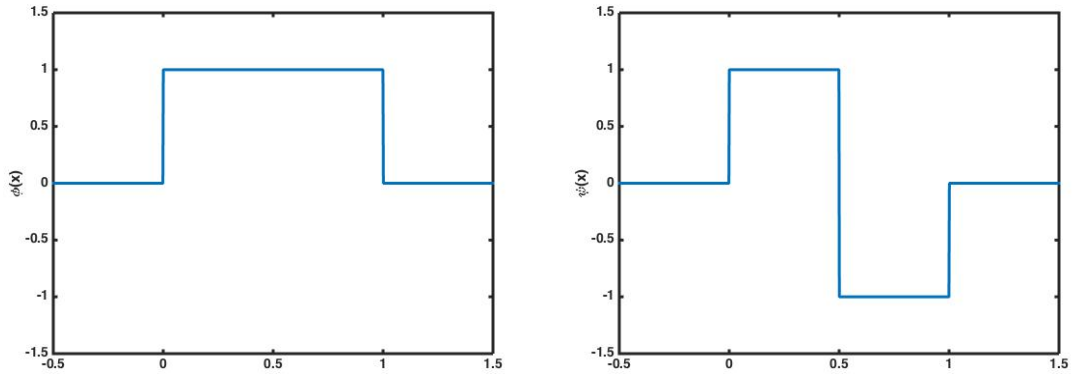

Figure 2.1: Left: (a) the Haar scaling function $\phi(x)$. Right: (b) the Haar wavelet $\psi(x)$. Both wavelets are defined along interval $[0, 1)$. Notice that the Haar wavelet is equivalent to the Haar step function along $[0, \frac{1}{2})$ minus the Haar step function along $[\frac{1}{2}, 0)$.

for a visual of the Haar wavelet. The scaling function ϕ and the wavelet ψ comprise the basic building blocks of all our wavelet transforms. In particular, it is important that the wavelet can be defined in terms of a step function, as this allows us to solve for our signal as a linear combination of ϕ and ψ in the same way that traditional Fourier analysis solves for the signal as a combination of sine and cosine functions. More concretely, ϕ and ψ represent local densities and local fluctuations, or local approximations and local differences, in the density field. We will explain this more in-depth in Section 2.2, where we discuss the actual transformation processes.

2.1.2 The Daubechies-2 Wavelet

We have chosen to present the Haar wavelet (sometimes referred to as the Daubechies-1 wavelet) because its simplicity will make an explanation of the transforms in the following sections much more intuitive. However, there are some obvious issues. For one, it is clearly a jump-discontinuous function and therefore any attempt to represent a signal with the Haar wavelet in frequency space will suffer

for it. We therefore introduce the next in the family of Daubechies wavelets, the Daubechies-2 wavelet.

Similar to the Haar wavelet, there are two closely related functions that we must explicate. The first is the scale function ϕ, which is not definable by explicit characterization but must rather be defined by recursion and initial conditions. Daubechies found this by setting [18]

$$\begin{aligned} \phi(0) &= 0 \\ \phi(1) &= \tfrac{1+\sqrt{3}}{2} \\ \phi(2) &= \tfrac{1-\sqrt{3}}{2} \\ \phi(3) &= 0 \end{aligned} \tag{2.5}$$

with the remaining points defined by the recursive equation

$$\phi(x) = h_0\phi(2x) + h_1\phi(2x-1) + h_2\phi(2x-2) + h_3\phi(2x-3) \tag{2.6}$$

where $\phi(x) = 0$ whenever $x < 0$ or $x > 3$. The four constants are

$$\begin{aligned} h_0 &= \tfrac{1+\sqrt{3}}{4} \\ h_1 &= \tfrac{3+\sqrt{3}}{4} \\ h_2 &= \tfrac{3-\sqrt{3}}{4} \\ h_3 &= \tfrac{1-\sqrt{3}}{4} \end{aligned} \tag{2.7}$$

These properties define a function that satisfies all the necessary conditions—oscillatory, square-integrable, localized, etc.—to be a wavelet. See Figure 2.2a for an illustration of the Daubechies-2 scaling function $\phi(x)$. Notice, crucially, that like the Haar step function and wavelet this function is localized to a finite interval, in this case $(0, 3)$.

Also similar to the Haar wavelet, we define the Daubechies-2 wavelet in terms of the scaling function

$$\psi(x) = -h_0\phi(2x-1) + h_1\phi(2x) - h_2\phi(2x+1) + h_3\phi(2x+2) \tag{2.8}$$

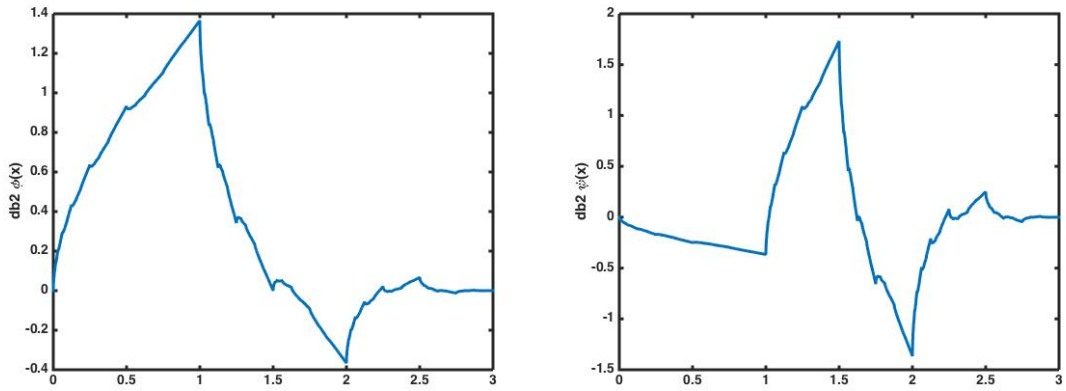

Figure 2.2: Left: (a) the Daubechies-2 scaling function $\phi(x)$. Right: (b) the shifted Daubechies-2 wavelet $\psi(x-1)$. Both are defined along interval $(0,3)$.

where the h-constants are the same as above. As the scaling function is defined by recursion, so too is the wavelet. Note, however, that while in the case of the Haar wavelet the step and wavelet functions were both nonzero over the same interval, Figure 2.2b depicts the Daubechies-2 wavelet *shifted* by 1 to the right; in fact, the Daubechies wavelet is nonzero over the interval $(-1,2)$, which is distinct from the interval of its scaling function, $(0,3)$. As we will be translating and stretching the function when we do our transforms, this doesn't matter much. However it is helpful to note this property, as it is a possible source of confusion.

Though it is more difficult to see why than in the case of the Haar wavelet, the same general principle applies to wavelets generally: once we have written our signal in terms of these functions, ϕ and ψ represent, respectively, local densities and local fluctuations in our signal.

2.2 Wavelet Transforms

In the previous section, we discussed wavelets, the class of functions that will be used in wavelet transforms. In this section, we will discuss two specific kinds of

wavelet transforms, the Discrete Wavelet Transform (DWT) and the Wavelet Packet Transform (WPT). At its core, both of these transforms exploit the fact that the scaling function can be expanded in terms of a stretched scaling function and a stretched wavelet function. Since we are using a signal that has been measured experimentally, we will need to do our transform on a discrete signal. The generic form of a transform on a discrete signal, then, can be written as

$$\sum_{j=1}^{2N} s_j \phi(x) \rightarrow \sum_{j=1}^{N} a_j \phi'(x) + \sum_{j=1}^{N} d_j \psi'(x) \tag{2.9}$$

where s_j, represents the experimental signal coefficients, and a_j, and d_j are coefficients of the wavelet and scaling functions which serve as a basis for our transform. Thus, we will first represent the signal to be analyzed as an expansion in terms of scaling functions, that is, in the form of the summation on the left in Equation 2.9

$$f(x) \rightarrow \sum_{j=1}^{2N} s_j \phi(x)$$

where $f(x)$ represents the underlying, continuous real signal and the s_j coefficients represent the approximation that an experimental signal makes of the real signal. After a brief explanation of signal approximation, we will explicate these transforms and give examples of how they both function in both the Haar and Daubechies-2 cases.

2.2.1 Signal Approximation

Haar Approximation Suppose we have a function $f(x)$ that we have sampled at even intervals x_j, with individual data points s_j

$$f(x_j) = s_j \tag{2.10}$$

for some set of integers j. We represent the signal we are going to transform as a vector

$$\vec{s} = (s_0, s_1, ..., s_k)$$

where the samples are taken at an even sampling frequency. For reasons that will become apparent later, this signal must have a length that is a power of two. Therefore, $k = 2^N - 1$ for some integer N. Our aim is to transform our signals into a form in which we can easily extract the sort of information we are looking for. The first step in doing this is to approximate our signal in terms of our scale functions, ϕ. Using the Haar wavelet,

$$\tilde{f}(x) = \sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x) \tag{2.11}$$

$\tilde{f}(x)$ is now the approximated signal, and $[x_j, x_{j+1})$ represents an interval over which s_j is sampled.

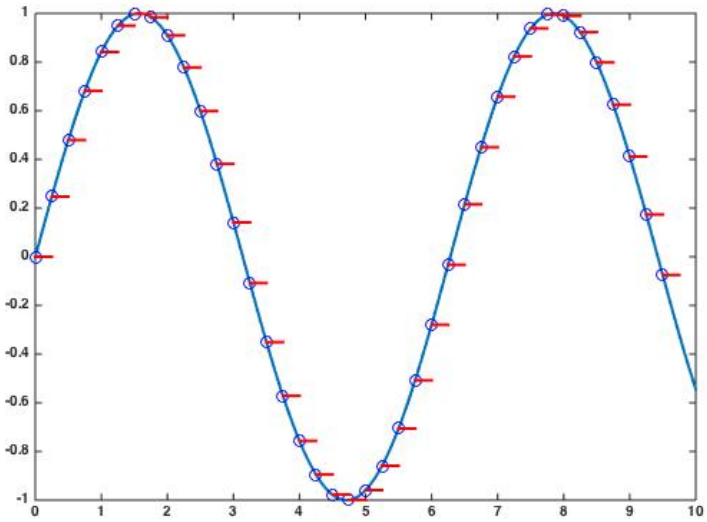

Figure 2.3: If the solid blue line is our signal $f(x)$ and the blue circles are the points at which the signal is sampled s_j, then the discontinuous red function represents the approximation of this signal $\tilde{f}(x)$ according to Equation 2.11.

In Figure 2.3, we have a graph of a signal $f(x)$ (the blue function), the sampled data points s_j (the blue dots along the function), and the approximation of our signal $\tilde{f}(x)$ given by Equation 2.10 (the red function). The Haar ϕ function is just a

step function, so this summation is simply a series of non-overlapping step functions spaced along the interval of the signal, one per data point. And because each step function is multiplied by the measured value of the signal in the corresponding interval, we know that this approximation interpolates our original function at the relevant points.

For example, we can compute the value of the function at a point x' using

$$\tilde{f}(x') = \sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x').$$

The intervals of the individual step functions in this summation do not overlap. Thus, we simply select the interval that contains x', and all other terms will be zero, such that

$$\tilde{f}(x') = s_k, \ x' \in [x_k, x_{k+1}).$$

As can be seen from Figure 2.3, the signal approximation interpolates the original signal exactly on the sample points x_j, and does a fair job of approximating the value of the original signal between these data points. In particular, because we wanted to illustrate the approximation, the example in Figure 2.3 has fewer sampling points than a real signal. A real signal, with many more data points, would be even better approximated.

Daubechies Approximation With the Daubechies wavelet, we still need to approximate our signal with an expansion of scaling functions. But because the Daubechies scaling function is more complex than the simple step Haar scaling function, this approximation is less straightforward. Once again, we start with our function, $f(x)$, which we sample at discrete points x_j, where $f(x_j) = s_j$.

To approximate our signal, we will exploit some of the convenient numerical properties of the Daubechies scaling function. If we solve the Daubechies scaling function

ϕ at the following three values of x

$$\phi(650/1024) = .999985$$

$$\phi(1 + 650/1024) = -.000155$$

$$\phi(2 + 650/1024) = .000170$$

we notice that the first is approximately equal to 1, while the remaining two are approximately equal to 0. Let $r_0 = 650/1024$; then, based on the three calculations above and the fact that the scaling function is zero outside the interval $[0, 3)$, we can conclude that for any integer n

$$\phi(r_0 + n) \approx \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n = 1,\ 2,... \end{cases} \tag{2.12}$$

Therefore, if f_s is the sampling frequency, the following summation will approximate our signal $f(x)$ at the points

$$\tilde{f}(x) = \sum_{k=0}^{2^N-1} s_k \phi(r_0 + f_s[x - x_k]) \tag{2.13}$$

Figure 2.4 shows us the Daubechies-2 approximation of the same signal function used in Figure 2.3. Unlike the Haar approximation, the approximation is smooth and continuous, and in general it is much more accurate.

With the Daubechies-2 approximation, there are some extra shift-terms with r_0 that must be taken into account; however, the important point is that we can represent our original function as a series of coefficients times the scaling function shifted along the domain. Generally, we will eliminate the extra factor of r_0 in our subsequent treatment. Our wavelet transforms will still be able to act on these approximations and, crucially, their coefficients. And because we are interested in our signal's frequency spectrum, this extra shift will not affect our final result.

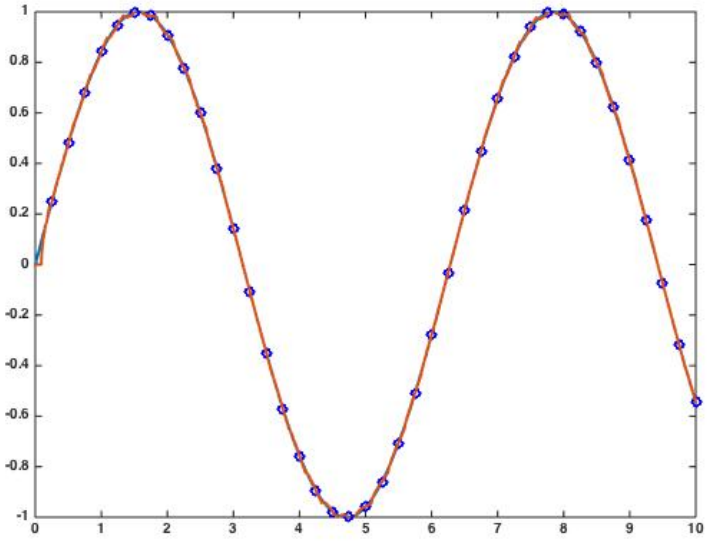

Figure 2.4: If the solid blue line is our signal $f(x)$ and the blue circles are the points at which the signal is sampled s_j, then the red function represents the approximation of this signal $\tilde{f}(x)$ according to Equation 2.13. As we can see, the approximation is extremely good, such that the blue line is barely visible underneath the red line.

2.2.2 The Discrete Wavelet Transform

The Haar DWT Now that we have an approximation of the signal and understand the basic properties of our wavelets, we can introduce our actual transform. Like Fourier analysis, our goal is to expand our original function in a particular basis. We'll begin with the Haar version of the transform, as this is simpler. Given the approximation of the signal given above in Equation. 2.11,

$$f(x) = \sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x)$$

the Haar wavelet transform replaces each successive pair of scaling functions with a wider scaling function and a wavelet. The wider scaling function 'course-grains' the signal, while the wavelet captures the fluctuations at that scale. As we have

discussed, the scaling function ϕ and the wavelet ψ represent local densities and local fluctuations, respectively. Thus, this mathematical process of rewriting a pair of scaling functions in terms of ϕ and ψ effectively represents the process of expanding the function in terms of a basis that represent these local densities and local approximations. Mathematically, we obtain these results by noting the following property of the Haar scaling and wavelet functions

$$\phi_{[a,b)}(x) = \phi_{[a,\frac{a+b}{2})}(x) + \phi_{[\frac{a+b}{2},b)}(x) \tag{2.14}$$

$$\psi_{[a,b)}(x) = \phi_{[a,\frac{a+b}{2})}(x) - \phi_{[\frac{a+b}{2},b)}(x) \tag{2.15}$$

We can take advantage of this relationship by inverting it

$$\phi_{[a,\frac{a+b}{2})} = \frac{1}{2}\left[\phi_{[a,b)} + \psi_{[a,b)}\right] \tag{2.16}$$

$$\phi_{[\frac{a+b}{2},b)} = \frac{1}{2}\left[\phi_{[a,b)} - \psi_{[a,b)}\right] \tag{2.17}$$

So that for each pair of terms in our approximation, we get

$$s_j\phi_{[x_j,x_{j+1})} + s_{j+1}\phi_{[x_{j+1},x_{j+2})} = \frac{s_j + s_{j+1}}{2}\phi_{[x_j,x_{j+2})} + \frac{s_j - s_{j+1}}{2}\psi_{[x_j,x_{j+2})} \tag{2.18}$$

We will call the coefficients of the new, wider step functions approximation coefficients a_j (because they approximate the average of the function at those two points), and coefficients of the wavelet function difference coefficients d_j (because they tell us the difference of the two points from that average). As a result, our approximation gives us

$$f(x) = \sum_{j=0}^{2^N-1} s_j\phi_{[x_j,x_{j+1})}(x) = \sum_{j=0}^{2^{(N-1)}-1} a_j\phi_{[x_{2j},x_{2(j+1)})}(x) + d_j\psi_{[x_{2j},x_{2(j+1)})}(x) \tag{2.19}$$

where our approximation and difference coefficients are defined as

$$a_j = \frac{s_{2j} + s_{2(j+1)}}{2} \tag{2.20}$$

$$d_j = \frac{s_{2j} - s_{2(j+1)}}{2} \tag{2.21}$$

This constitutes a single 'pass' of the DWT. To understand what we have accomplished here, consider by analogy Fourier analysis. In Fourier analysis, the original signal function is expanded in a basis of sine and cosine waves, which allow us to analyze the signal in frequency space. In the first pass of the DWT, the transform has written our function as an expansion of local approximations a_j and local differences d_j.

From our original signal $\vec{s}$ we have derived a way to calculate approximation and difference vectors. A few things are important to note. First, this transformation is completely reversible; no information is lost when we calculate our approximation and difference coefficients. Indeed it would be a simple matter to write expressions that inverted the transform

$$s_{2j} = 2(a_j + d_j)$$

$$s_{2j+1} = 2(a_j - d_j)$$

Second, we can now write our signal as

$$\vec{s}^1 = (a_0, a_1, ..., a_{2^{(N-1)}-1}; d_0, d_1, ..., d_{2^{(N-1)}-1}) \tag{2.22}$$

All of this constitutes the 'first pass' of the DWT. Once we have our coefficients from the first pass, the DWT involves making a second pass on the signal, treating the new approximation coefficients as the new signal. To distinguish between approximation and detail coefficients arising from the first and successive passes, we will use the notation a_i^n and d_i^n for the approximation and difference coefficients resulting from the nth pass. The generalized relationship between coefficients of subsequent passes of the DWT is

$$a_j^{n+1} = \frac{a_{2j}^n + a_{2(j+1)}^n}{2} \tag{2.23}$$

$$d_j^{n+1} = \frac{a_{2j}^n - a_{2(j+1)}^n}{2} \tag{2.24}$$

Figure 2.5 illustrates the relationship between the various levels of the DWT for a signal which undergoes 3 passes. With each successive pass of the DWT, the

number of approximation coefficients is reduced by two, with an equal number of difference coefficients. For a longer signal, additional passes are added, but the overall structure remains the same.

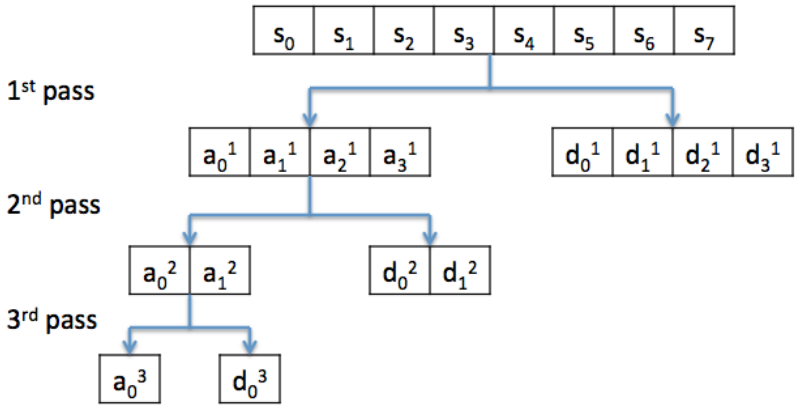

Figure 2.5: Tree representing the relationship between 3 levels of the DWT.

Generally, after N passes (on a signal that is 2^N data points long), our signal becomes

$$\vec{s}^N = (a_0^N; d_0^N; d_0^{N-1}, d_1^{N-1}; ...; d_0^1, d_1^1, ..., d_{2^{(N-1)}-1}^1) \tag{2.25}$$

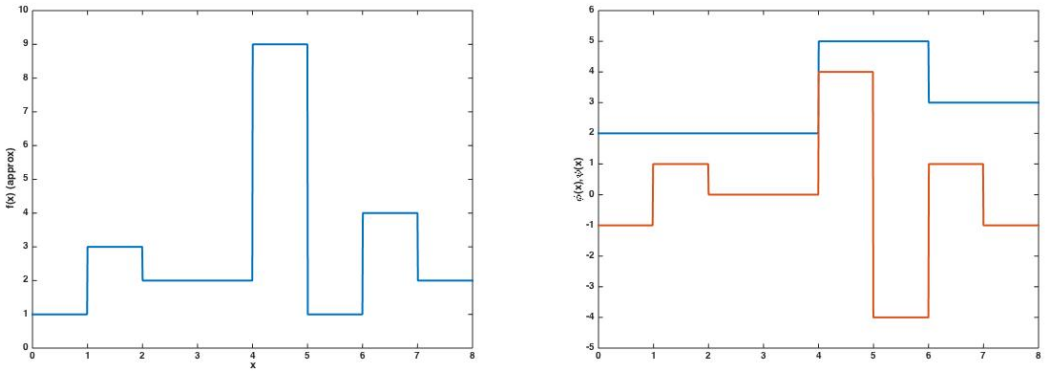

Figure 2.6: Left: (a) an example signal. Right: (b) the signal separated into functions representing its local approximations (the blue line) and its local fluctuations (the red line).

To illustrate this method, we will do a simple example. Take the signal

$$\vec{s} = (1, 3, 2, 2, 9, 1, 4, 2)$$

The first pass of the DWT on this signal is illustrated in Figure 2.6. For simplicity, assume the sampling frequency is 1, starting with zero, so that the appropriate approximation of our signal (shown in Figure 2.5a) is

$$f(x) = \sum_{j=0}^{7} s_j \phi_{[j,j+1)}$$

$$f(x) = 1\phi_{[0,1)} + 3\phi_{[1,2)} + 2\phi_{[2,3)} + 2\phi_{[3,4)} + 9\phi_{[4,5)} + 1\phi_{[5,6)} + 4\phi_{[6,7)} + 2\phi_{[7,8)}$$

The first pass of the wavelet transform transforms each individual pair of coefficients into an approximation and difference coefficient,

$$a_0^1 = \frac{s_0 + s_1}{2} = \frac{1+3}{2} = 2$$

$$d_0^1 = \frac{s_0 - s_1}{2} = \frac{1-3}{2} = -1$$

$$1\phi_{[0,1)} + 3\phi_{[1,2)} = 2\phi_{[0,2)} + -1\psi_{[0,2)}$$

And so on, so that our signal can be rewritten as

$$f(x) = 2\phi_{[0,2)} + -1\psi_{[0,2)} + 2\phi_{[2,4)} + 0\psi_{[2,4)} + 5\phi_{[4,6)} + 4\psi_{[4,6)} + 3\phi_{[6,8)} + 1\psi_{[6,8)}$$

We can then rearrange these terms so that all of the ϕ functions are grouped with each other, and likewise all the ψ functions,

$$f(x) = 2\phi_{[0,2)} + 2\phi_{[2,4)} + 5\phi_{[4,6)} + 3\phi_{[6,8)} + -1\psi_{[0,2)} + 0\psi_{[2,4)} + 4\psi_{[4,6)} + 1\psi_{[6,8)}$$

The first four terms correspond to the blue function in Figure 2.6b, and the latter four terms correspond to the red function in Figure 2.6b. Physically, we interpret the first grouping—which corresponds to the approximation coefficients in the transform—as a course-graining of the signal. The latter group is the fluctuations

lost when we course-grain the signal; as can easily be checked, adding these two functions would return the original signal function. These difference coefficients encode information about the original signal's frequency spectrum.

Now that we have transformed the signal, it is expedient to group the coefficients so that

$$\vec{s_1} = (2, 2, 5, 3; -1, 0, 4, 1)$$

where the first four numbers correspond to the coefficients of the ϕs, and the second four numbers correspond to the coefficients of the ψs. What we have accomplished thus far represents the first pass of the DWT.

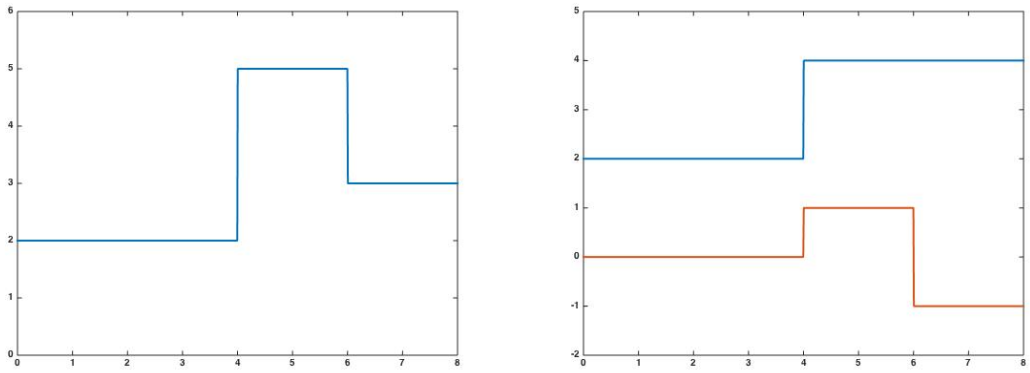

Figure 2.7: The second pass of the Haar DWT. Left: (a) the course-grained signal (carried over from Figure 2.6b). Right: (b) the course-grained signal separated into functions representing its local approximations (the blue line) and its local fluctuations (the red line).

To continue, we perform a second pass *on the approximation coefficients only.* Figure 2.7a shows the transition from the course-grained signal, carried over from Figure 2.6b, to the results of the second pass of the DWT. The four last terms are carried over unchanged, but the initial two terms undergo transformations according to Equations 2.23 and 2.24,

$$a_0^2 = \frac{a_0^1 + a_1^1}{2} = \frac{2+2}{2} = 2$$

$$d_0^2 = \frac{a_0^1 - a_0^1}{2} = \frac{2-2}{2} = 0$$

$$2\phi_{[0,2)} + 2\phi_{[2,4)} = 2\phi_{[0,4)} + 0\psi_{[0,4)}$$

And so on, so that our signal becomes

$$f(x) = 2\phi_{[0,4)} + 4\psi_{[0,4)} + 0\phi_{[4,8)} + 1\psi_{[4,8)} + [-1\psi_{[0,2)} + 0\psi_{[2,4)} + 4\psi_{[4,6)} + 1\psi_{[6,8)}]$$

So that, after grouping the approximation coefficients and difference coefficients once more, our next signal vector is

$$\vec{s}_2 = (2, 0; 4, 1; -1, 0, 4, 1)$$

And, finally, we can make one final pass

$$f(x) = 1\phi_{[0,8)} + 1\psi_{[0,8)} + 0\psi_{[0,4)} + 1\psi_{[4,8)} + -1\psi_{[0,2)} + 0\psi_{[2,4)} + 4\psi_{[4,6)} + 1\psi_{[6,8)}$$

and our final signal vector

$$\vec{s}_3 = (1; 1; 0, 1; -1, 0, 4, 1)$$

This is the final result of the full Haar DWT.

The Daubechies DWT The Daubechies-2 DWT is very similar to the Haar DWT—everything that we have said about splitting our signal into approximation and difference coefficients holds, as well as the method of making successive passes on our signal with our transform method. Indeed, the general tree in Figure 2.5 is generally true of all DWT, regardless of the wavelet used. However, the transition from our signal coefficients to approximation and difference coefficients is a little more complex, and Equations 2.23 and 2.24, which we derived from the form of the Haar wavelet and scaling function, will no longer apply. For the Daubechies-2 DWT, we need to use the relationship between the Daubechies-2 wavelet and scaling function.

Thus, if we substitute $r/2$ for x in the original recurrence relations that we used to generate the Daubechies-2 scaling function and wavelet function (Equations 2.6 and 2.8, using the coefficients from Equation 2.7), we get an equation for larger, 'stretched' wavelets in terms of smaller scaling functions.

$$\phi(r/2) = h_0\phi(r) + h_1\phi(r-1) + h_2\phi(r-2) + h_3\phi(r-3) \tag{2.26}$$

$$\psi(r/2) = -h_0\phi(r-1) + h_1\phi(r) - h_2\phi(r+1) + h_3\phi(r+2) \tag{2.27}$$

By incorporating the necessary shift and rearranging terms,

$$\phi(r/2-k) = h_0\phi(r-2k)+h_1\phi(r-1-2k)+h_2\phi(r-2-2k)+h_3\phi(r-3-2k) \tag{2.28}$$

$$\psi(r/2-1-k) = h_3\phi(r-2k)-h_2\phi(r-1-2k)+h_1\phi(r-2-2k)-h_3\phi(r-3-2k) \tag{2.29}$$

In the same way that the Haar DWT separated the smaller scaling functions into a basis of larger scaling functions and wavelets, so too does the Daubechies-2 DWT. Unlike the Haar DWT, each Daubechies-2 DWT coefficient depends on four coefficients.

If we start once more with our signal vector $\vec{s}$, the overall transform can be represented by a matrix multiplication

$$\begin{pmatrix} a_0^1 \\ d_0^1 \\ a_1^1 \\ d_1^1 \\ a_2^1 \\ d_2^1 \\ \vdots \end{pmatrix} = \frac{1}{2} \begin{pmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & \cdots \\ h_3 & -h_2 & h_1 & -h_0 & 0 & 0 & \cdots \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & \cdots \\ 0 & 0 & h_3 & -h_2 & h_1 & -h_0 & \cdots \\ 0 & 0 & 0 & 0 & h_0 & h_1 & \cdots \\ 0 & 0 & 0 & 0 & h_3 & -h_2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ \vdots \end{pmatrix} \tag{2.30}$$

Note that coefficient in the first row of the resultant column, a_0^1, depends only on the first four entries in the signal vector—and the corresponding difference coefficient,

d_0^1, depends on the same four coefficients. The next approximation coefficient then *skips* two signal points, and depends only on s_2 through s_5, and so on. Just as in the case of the Haar DWT, the approximation coefficients are used to give a course-grained signal, while the difference coefficients capture the fluctuations at given scales. As well, it is a simple matter to invert this transform—we simply take the inverse of the transformation matrix.

The vector of coefficients on the left, properly reordered such that we group the approximation and difference coefficients, is the transformed signal vector for the first pass of the DWT. Like the Haar DWT, we treat the group of approximation coefficients as the new signal and make a successive passes on these using the same matrix method.

2.2.3 The Wavelet Packet Transform

The WPT is remarkably similar to the DWT. In fact, it consists of an extension of the DWT's methods. The basic method for acquiring approximation and difference coefficients remains the same, so there is no need to repeat it here. The difference, however is this: where the DWT only makes a successive passes on the approximation coefficients, the WPT extension enacts successive transforms on both approximation *and* difference coefficients. See Figure 2.8 for an example of the WPT tree, and compare this to Figure 2.5, the DWT tree. It is largely the same; the only major difference is that the branches with difference trees do not terminate. As a result, we end up with more information about our signal in the frequency domain.

To keep track of what is an approximation coefficient and what is a difference coefficient at various levels, we will use the following recursive convention. For a given coefficients with names of the form $xy...z_i$, where x, y, z are some combination of a's and d's and i are the numbers that differentiate these coefficients, $xy...za_i$

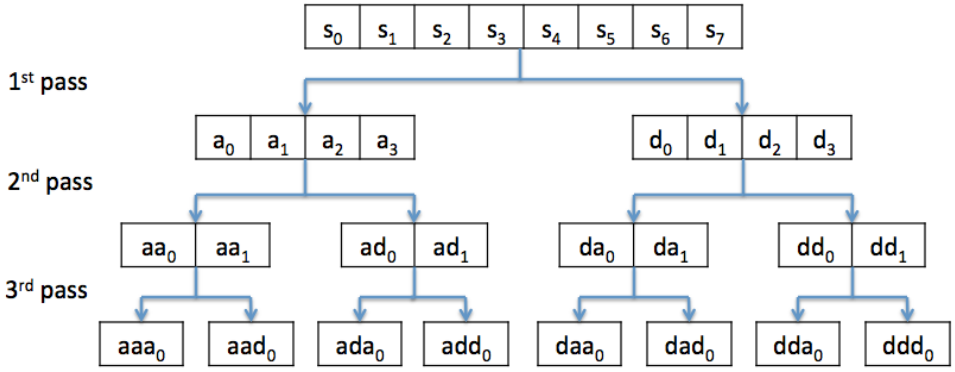

Figure 2.8: Tree representing full Haar WPT.

will be the approximation coefficients generated by performing the transform on the $xy...z_i$ coefficients, and $xy...zd_i$ will be the corresponding difference coefficients. Thus, a_i and d_i are the approximation and difference coefficients from the first pass of the transform, while aad_i are the difference coefficients generated by acting on the aa_i coefficients, which are in turn the approximations coefficients generated by performing the transform on the original approximation coefficients a_i.

As before, the approximation and difference coefficients of each individual pass of the transform separate out the local densities and local fluctuations. For the WPT, however, the easiest way to interpret the coefficients after successive transforms is in terms of high- and low-pass filters—at each level, the approximation coefficients are the result of the action of a low-pass filter on the signal, while the difference coefficients are the result of the action of a high-pass filter on the signal. Thus, for example, the a_i coefficients contain information about low-frequency band of the original signal, while the ad_i coefficients contain information about the higher end of the low-frequency band of the original signal, and so forth.

As an example, we will do a Haar WPT on the same example signal as above. Figure 2.8 tracks the progress of the WPT on our sample signal, while Figure 2.9 is the

same as Figure 2.8 with the relevant details from our example filled in.

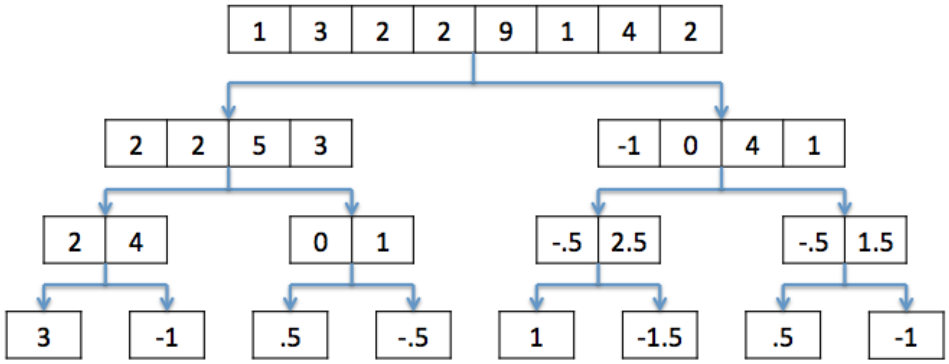

Figure 2.9: Tree representing the Haar WPT on our sample signal.

$$\vec{s} = (s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7) = (1, 3, 2, 2, 9, 1, 4, 2).$$

After the first pass of the WPT, the our output vector of coefficients is identical to the one we obtained when doing the first pass of the DWT,

$$\vec{s}^1 = (a_0, a_1, a_2, a_3; d_0, d_1, d_2, d_3) = (2, 2, 5, 3; -1, 0, 4, 1).$$

Successive passes not only act on our approximation coefficients, but also on our difference coefficients. Note that the action of the WPT is the same for the first four coefficients, but *also* acts on the last four coefficients.

$$\vec{s}^2 = (aa_0, aa_1; ad_0, ad_1; da_0, da_1; dd_0, dd_1) = (2, 4; 0, 1; -.5, 2.5; -.5, 1.5).$$

Finally, we are left with a vector of completely transformed coefficients. Note that this transform also requires us to have a signal length of some power of two,

$$\vec{s}^3 = (aaa_0; aad_0; ada_0; add_0; daa_0; dad_0; dda_0; ddd_0) = (3; -1; .5; -.5; 1; -1.5; .5; -1)$$

2.3 2-D Transforms

We have reviewed the DWT and the WPT for 1-dimensional signals. However, we will need to apply our analysis to 2-dimensional fields; hence, we will need to extend our transform methods.

The process in going from 1-D to 2-D is straightforward. Given a signal (now in the form of a 2^N by 2^N matrix), we can take advantage of the orthogonality of the wavelet transform and perform the one-dimensional transform twice: first, on each row of our matrix, second, on each column. Take, for example, this 2-d sample:

$$\left\{ \begin{array}{cc} s_{0,0} & s_{0,1} \\ s_{1,0} & s_{1,1} \end{array} \right\} \tag{2.31}$$

We could perform the 1-d transform on rows or columns first—either way, we will obtain the same result. We will perform it on the rows first

$$\left\{ \begin{array}{cc} \frac{s_{0,0}+s_{0,1}}{2} & \frac{s_{0,0}-s_{0,1}}{2} \\ \frac{s_{1,0}+s_{1,1}}{2} & \frac{s_{1,0}-s_{1,1}}{2} \end{array} \right\} \tag{2.32}$$

And then, on the columns

$$\left\{ \begin{array}{cc} \frac{(s_{0,0}+s_{0,1})+(s_{1,0}+s_{1,1})}{4} & \frac{(s_{0,0}-s_{0,1})+(s_{1,0}-s_{1,1})}{4} \\ \frac{(s_{0,0}+s_{0,1})-(s_{1,0}+s_{1,1})}{4} & \frac{(s_{0,0}-s_{0,1})-(s_{1,0}-s_{1,1})}{4} \end{array} \right\} \tag{2.33}$$

In this case, we have four types of coefficients: the approximation coefficients a_i, the diagonal difference coefficients d_i, the horizontal difference coefficients h_i, and the vertical difference coefficient v_i, all placed such that

$$\left\{ \begin{array}{cc} a_i & h_i \\ v_i & d_i \end{array} \right\} \tag{2.34}$$

Like the one-dimensional case, we can group coefficients of the same type and make a successive passes. Like the one dimensional case, we will only make passes on

the approximation coefficients for the 2-d DWT, and we will make passes on all the different kinds of coefficients for the 2-d WPT. Figure 2.10 is a schematic of the tree for a 2 dimensional WPT; additional passes of the WPT follow the same general pattern. There are a couple of important distinctions between the 1- and 2-dimensional WPT that are worth highlighting.

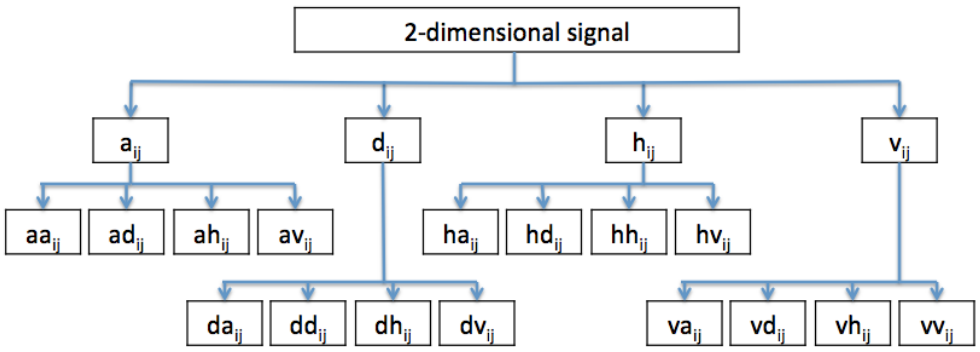

Figure 2.10: Tree representing the 2 dimensional WPT. Unlike the case of the one-dimensional signal, each leaf of the WPT represents a matrix of coefficients, instead of merely a vector of coefficients.

First, because we have a two-dimensional signal represented by a matrix instead of a vector, the individual sets of coefficients are also matrices instead of vectors. In Figure 2.10, this is represented by indexing the coefficients by ij instead of merely i. Second, because of the way that the 2-dimensional WPT transforms coefficients, the number of coefficients of each type after the transform is reduced by a factor of 4 instead of by a factor of 2.

2.4 Wavelet vs. Fourier Methods

We are going to use these wavelet analysis techniques to examine a 2-dimensional signal in frequency space, i.e., to calculate the power spectrum as explained in Chapter 1. Of course, wavelets are not the only method by which we can do spectral

analysis—a widely-used alternative method is Fourier analysis. Thus, in this section we will explain some of the advantages and disadvantages to using wavelet methods over Fourier methods.

The most obvious benefit associated with wavelets is their localization. Fourier waves span over an infinite domain, and therefore in a Fourier transform all localization information is lost. For example, when one does a Fourier analysis of a sound wave, we obtain information on the what the notes/frequencies are in the sound sample, but not when in the sample they occurred. Wavelets are nonzero only over a particular interval, and one can therefore use wavelets to investigate the properties of a signal in an interval which *excludes* other parts of the signal.

This localization property has a number of helpful side effects. First, because our wavelet basis is orthogonal and compact (i.e., only nonzero over a finite interval), we do not need to worry about aliasing. Second, because the transform is always performed locally, any bad data points will only infect a single branch of our packet tree; compare this to a Fourier transform, where a single bad data point forces every term in the Fourier series transform to accommodate it.

The primary disadvantage to using wavelet methods is the issue of wavelet scale to characteristic frequency; where a Fourier analysis expands a function in terms of sine and cosine waves which have an obvious natural frequency, wavelets do not have an obvious natural frequency (See Fig. 2.1 and 2.2; while both look roughly like a wave, it is clearly not an exact match). Thus, when transforming a function using wavelets and using their coefficients for frequency analysis we must also ask what frequency these wavelets correspond to. This will be discussed further in Chapter 3, when we discuss details of our spectral analysis methods.

CHAPTER 3

Spectrum Estimation

In this chapter, we will lay out our implementation of the WPT in finding the power spectrum. In Section 3.1, we provide an overview of the various steps involved in our process. In Section 3.2, we will outline the aspects of the SDSS data which are of significance for our research and describe the process used to prepare data for analysis. In Section 3.3, we will detail the methods used to obtain the 1-dimensional power spectrum from our 2-dimensional DWT.

3.1 Procedure Overview

Figure 3.1 is a flowchart of our whole process; in it, ovals represent the various data inputs and outputs, while the boxes represent the various procedures used to process the data along the way. We begin with the SDSS galaxy data, described in Sections 3.2.2 and 3.2.3. This data is processed into a two-dimensional matrix signal by SDSSPix, described in Section 3.2.4. This data is subjected to the wavelet packet transform described in Chapter 2, and using the methods of spectrum estimation described in Section 3.3 the results of the WPT are converted into the 2-dimensional power spectrum. By a process of radial averaging described in Section 3.4, this 2-dimensional power spectrum is converted into a 1-dimensional power spectrum. Finally, by comparing this power spectrum to the fiducial cosmological data and previous BOSS results using the analysis methods described in Section 4.1, we extract cosmological information about the BAO from the power spectrum.

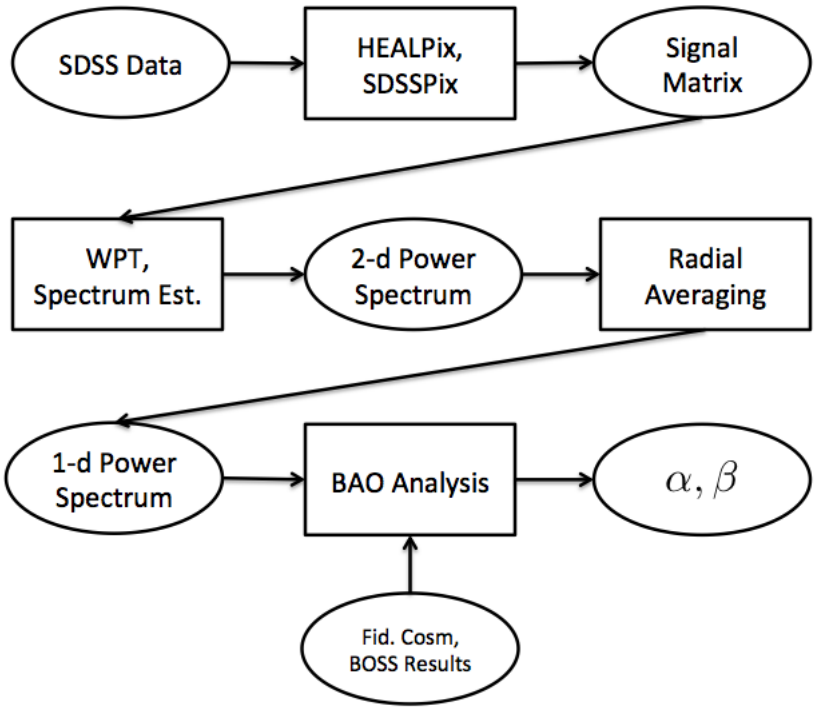

Figure 3.1: An overview flowchart of our entire process. Ovals in the flowchart represent data inputs/outputs, while rectangles represent the various processes and blocks of code that process data.

3.2 Preparing the Data

3.2.1 SDSS Instrumentation

To understand how we process the data provided by SDSS, we need to outline some of the details regarding SDSS instrumentation.

To obtain spectroscopic data of more than one target object at a time, the SDSS makes use of 'plug plates.' These are aluminum plates, approximately 32 inches in diameter, in which more than 1000 holes are strategically drilled.[1] Each hole

[1] A non-technical video of the plate production process can be found at https://youtu.be/iYyO7pGaJNw

corresponds to a target object, and when the plate is affixed to the SDSS 2.5 meter telescope, 62″ fiber-optic cables are plugged into each hole in the plate. During telescope operation, these fiber-optic cables then collect light from the target source and carry it to a spectroscopic analyzer. In this way, the telescope can collect spectroscopic data on each of the > 1000 objects, each centered at the focus of an individual hole in the plate. [9]

3.2.2 Celestial Coordinates

In the raw galaxy catalogues, the location of individual galaxies in the celestial sphere is given by the standard coordinates of right ascension (RA) and declination (DEC). However, because of the specifics of the SDSS survey, the patch of the night sky containing the sampled galaxies in the southern Galactic Cap straddles the RA $= 0° = 360°$ line.

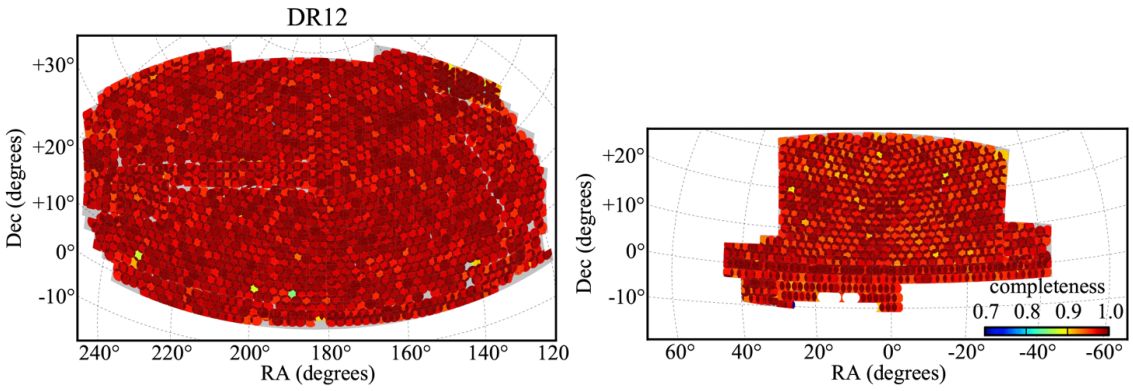

Figure 3.2: A depiction of SDSS coverage as of DR12, with standard RA-DEC celestial coordinates. Left: (a) the Northern Galactic Cap. Right: (b) the Southern Galactic Cap.

Note that in Figure 3.2b, the data data straddles the $RA = 0°$ line; since we require our RA data to be reported in the range of $[0°, 360°)$, any attempt on our part to project the sphere onto a square matrix using the standard HEALPix or SDSSPix

methods would result in our sample being split up and discontinuous. To compensate for this, we convert to a special set of celestial coordinates developed specifically for the SDSS survey. These coordinates are called LAMBDA (λ) and ETA (η), and are similar to the standard RA-DEC coordinates—the key difference being that the sample patch of galaxies is not discontinuous under these new coordinates. The formulas for conversion between RA-DEC and LAMBDA-ETA, where both are given in terms of degrees, are

$$x = \cos(ra - node)\cos(dec)$$

$$y = \sin(ra - node)\cos(dec)$$

$$z = \sin(dec)$$

$$\lambda = -\arcsin(x) \tag{3.1}$$

$$\eta = \arctan(z/y) - etaPole \tag{3.2}$$

where $node = 32.5$ and $etaPole = 95.0$ are constants, and $\arctan(y/z)$ is specifically the 4-quadrant arctan function, not merely the 2-quadrant one. Because we are working under the assumption of isotropy, this shift in coordinates will not have any effect on our results.

3.2.3 Weighting

In an ideal survey, we would have a list of objects, and each one would be counted exactly once. That is, each data point would be equally reliable. In the real world, the physical limitations of the survey equipment, the computational limitations of the data processing pipeline, and the noise produced by background effects introduce errors that require us to compensate for varying levels of confidence among data points and biases in our sampling. In this section, we will outline the various weighting parameters that are assigned to each galaxy in the SDSS survey and detail how they are finally combined into the total weighting parameter.

Close-Pair Weight The spectroscopic fiber-optic cables used in the SDSS telescope are of a finite width, specifically, $62''$. Thus, when two objects are closer than $62''$ in the night sky, they cannot be resolved. To account for this, if a given object is allocated a fiber, then its close pair weight (default $w_{cp} = 1$) is up-weighted by 1 for each target object within $62''$ of position O which is *not* allocated a fiber. This prevents the survey from significantly undersampling objects in small-scale clusters. [14]

Redshift Failure Weight For a given object, as part of its data processing pipeline, SDSS determines the object type (galaxy, quasar, etc.) and redshift. Because the methods for determining redshift are not 100% effective, there are sometimes errors in the measured values; for instance, there might be two distinct values of z for which the spectroscopic data is a significantly good match. A full list of these redshift failures is provided in [9]. When this happens, the object is marked with a 'redshift failure' warning. To account for this, redshift failure weight (default $w_{zf} = 1$) of the nearest object for which the redshift measurement was successful is up-weighted by 1. As an additional note, in some SDSS papers the redshift failure weight is denoted w_{rf}, but we will use w_{zf} in this document. [14]

Stellar Density Weight On the basis of research done in [19], it was found that the density of stars in a given region of the sky caused significant fluctuations in the observed density of galaxies in the CMASS galaxy sample. To compensate for this, a function of both stellar density and the surface brightness of the galaxy was introduced, w_{star}. This weight compensates for the bias introduced by varying stellar densities [1]. Since this bias is not present in the lower-redshift LOWZ galaxy catalogues, $w_{star} = 1$ for those galaxies [1].

Seeing Weight Lastly, on the basis of research done in [20], [21], and [19], SDSS weights galaxies according to other systematic effects—for example, airmass density—that affect the observed density of galaxies. These are grouped under the category of 'seeing' effects [1].

Total Weight The weights that we have outlined above are then combined into a single total weight, w_{tot}, according to [1].

$$w_{tot} = (w_{cp} + w_{zf} - 1)w_{star}w_{see} \tag{3.3}$$

Hence, rather than having an integer count of galaxies, the weighting procedure generally produces non-integer weights that measure how well we detected our objects and compensate for any biases that arise because of the limitations of the instrumentation, properties of the objects in the survey field, and atmospheric conditions at the time of representation.

3.2.4 HealPix and SDSSPix

Having established our celestial coordinates and weights, we now proceed to project our sample—naturally on the curved spherical surface of the celestial sphere—onto a flat matrix for analysis. For this purpose, we use SDSSPix, a variant of the more standard HealPix, developed specifically for SDSS.

HEALPix (and by extension SDSSPix) divide the sky into equal-area pixels. This is important because any bias in the size of area of the sky represented per pixel would skew the number density of galaxies. Based on the desired resolution, the base pixels (12 for HealPix, 468 for SDSSPix) are split into smaller pixels, as seen in Figure 3.3a—which show the HEALPix basis sphere being split at resolutions 2, 4, and 8. These pixels are then indexed according to a specific schema; a number of these different schemas are shown in Figure 3.3b. By using a pixel schema such

as one of those in Figure 3.3b, a patch of data on the surface of a sphere can be projected onto a plane in the form of a matrix. [22]

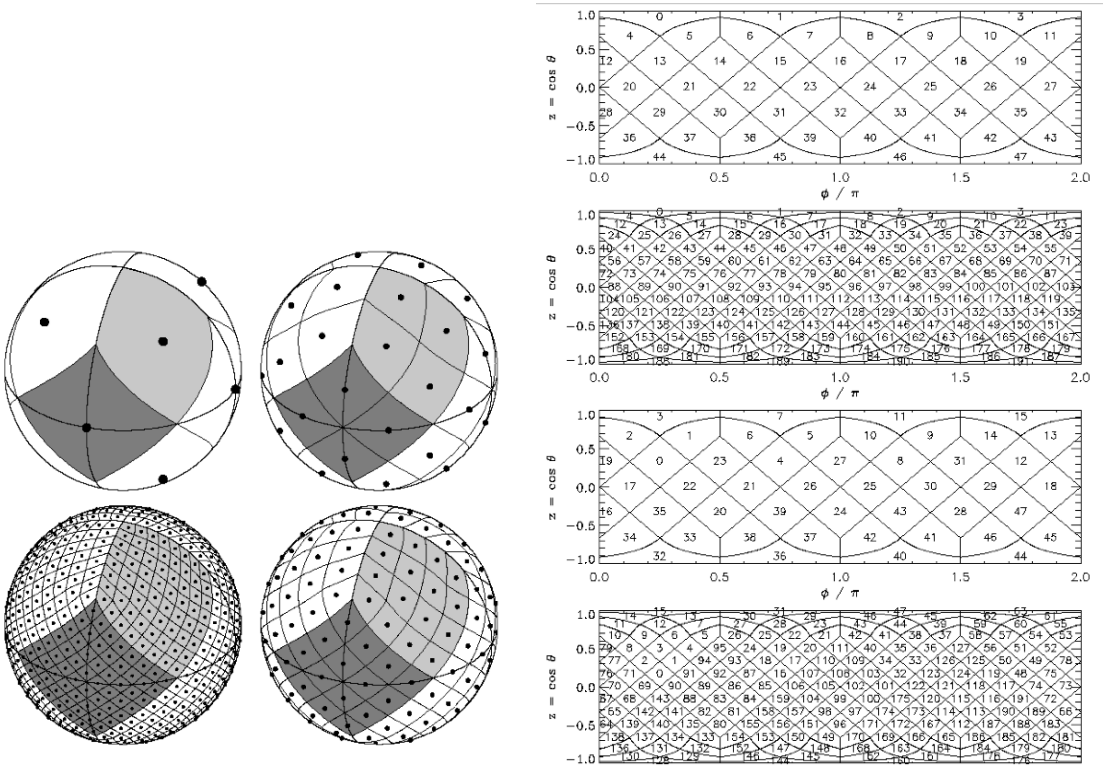

Figure 3.3: From [22]. Left: (a) A schematic representation of the HEALPix pixelation scheme on a sphere, at various resolutions: upper-left represents the base HEALPix pixelization scheme; proceeding clockwise, resolutions 2, 4, and 8. Right: (b) The HEALPix numbering schemes for ring (top, second) and nested (third, bottom) pixelization schemes.

The formula for converting from celestial coordinates (in radians) to pixel index in SDSSPix is

$$
\begin{aligned}
i &= \left\lfloor \frac{n_x * \eta}{2\pi} \right\rfloor \\
j &= \left\lfloor n_y * \frac{1 - \cos(\lambda)}{2} \right\rfloor \\
h_{index} &= n_x * j + i
\end{aligned}
\tag{3.4}
$$

where λ and η are the celestial coordinates defined in Equations 3.1 and 3.2, respectively, and n_x and n_y are the resolution times the number of base pixels in the x and y directions, respectively. Based on the standard SDSSPix layout, we convert the pixel index to a matrix location (r, c)

$$c = \lfloor h_{index}/n_x \rfloor + 1 \tag{3.5}$$

$$r = [h_{index}]_{n_x} + 5 * res \tag{3.6}$$

Given the matrix location of our data point, we then add w_{tot} for that galaxy to the appropriate matrix element.

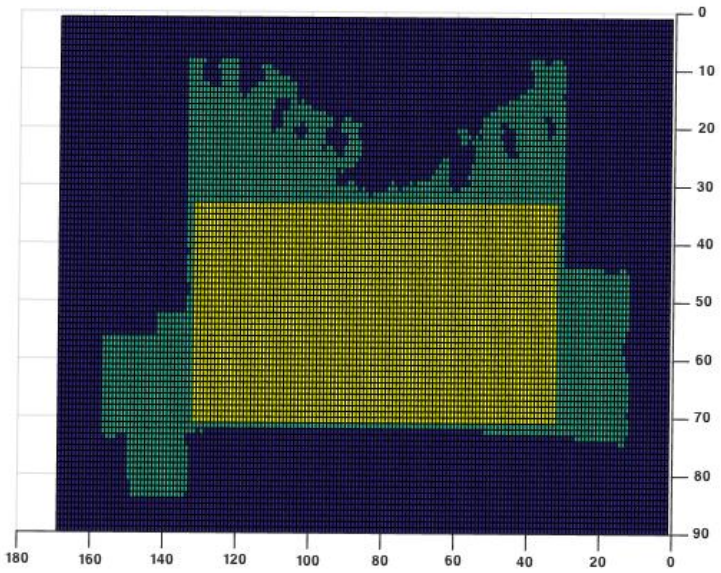

Figure 3.4: The SDSSPix rendering of the SDSS data in the Southern Galactic Cap at resolution 16. The green pixels are those which the SDSS survey covers, while the yellow represents the largest rectangular section within the available data.

For the purposes of the wavelet packet transform, we need a square matrix that is $2^n \times 2^n$, for some integer n. Out of the full matrix, we cut out the largest possible $2^n \times 2^n$ matrix which is completely contained inside the area of the sample patch.

Figure 3.4 is a cutout of the full matrix; the green indicates pixels where we have data. The yellow rectangle is the largest possible square matrix within the confines of the available data area, and from this we can cut out the largest possible $2^n \times 2^n$ square as our signal matrix for analysis. If there are multiple options for cutting out this largest possible square matrix, we simply choose at at random.

3.3 Producing the Power Spectrum

Using the $2^n \times 2^n$ matrix acquired as described in the previous section, we apply the 2-dimensional wavelet packet transform (WPT) described in Chapter 2. In this section, we will justify our use of the wavelet packet transform to estimate the power spectrum and describe the process by which we use the coefficients of this transform to estimate the two-dimensional power spectrum.

3.3.1 Parseval's Theorem

We are using wavelet methods to estimate the power spectrum. However, the most common method to analyze the frequency spectrum of a signal is the Fourier method. If the Fourier expansion of a signal of length L is given by

$$f(x) = \sum_{n=-\infty}^{\infty} \epsilon_n e^{i2\pi nx/L} \tag{3.7}$$

with coefficients given by

$$\epsilon_n = \frac{1}{L} \int f(x) e^{-i2\pi nx/L} dx \tag{3.8}$$

then Parseval's theorem for the Fourier transform is given by [23]

$$\frac{1}{L} \int_0^L |f(x)|^2 \, dx = \sum_{n=-\infty}^{\infty} |\epsilon_n|^2 \tag{3.9}$$

This is important for two reasons. First, this establishes that power is conserved under the Fourier transform. This makes the Fourier transform a legitimate means to analyze the power spectrum, as a transformation that did not conserve power could not possible be a good estimate of power. Second, it relates the coefficients of the Fourier transform, which are indexed according to frequency, to the power of individual frequency components in the signal, so that we can estimate the power spectrum as

$$P(n) = |\epsilon_n|^2 \tag{3.10}$$

To use wavelet transforms to estimate the power spectrum, we first need to meet the same criteria as the Fourier transform—that is, we must establish that the wavelet transforms preserve power. To do this, we will show that the transform preserves power at each pass, and this will inductively show that the power is preserved for a transform at any depth.

We begin with our signal estimate, which as was shown in section 2.2.1. is always in terms of generic scaling functions,

$$\tilde{f}_i(x) = \sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x)$$

And the first pass of the transform takes this series to a series of stretched scaling functions and wavelets,

$$\sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x) \xrightarrow{WPT} \tilde{f}_f(x) = \sum_{j=0}^{2^{N-1}-1} a_j \phi_{[x_{2j}, x_{2(j+1)})}(x) + \sum_{j=0}^{2^{N-1}-1} d_j \psi_{[x_{2j}, x_{2(j+1)})}(x)$$

To show that this transform preserves power, we need to show that

$$\int |\tilde{f}_i(x)|^2 \, dx = \int |\tilde{f}_f(x)|^2 \, dx \tag{3.11}$$

This is indeed true; see Appendix A for an explicit proof [23].

3.3.2 Spectrum Estimation

One-Dimensional Spectrum Estimation To understand the methods that we use in spectrum estimation, we need to understand the WPT from the perspective of signals analysis.

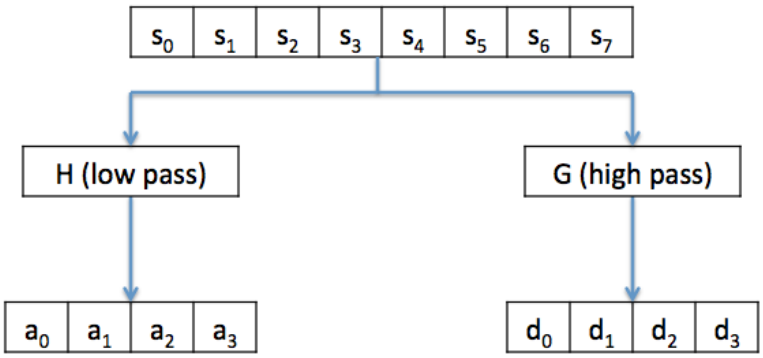

Figure 3.5: A representation of a pass of the WPT as the action of a low-pass filter (H) and a high-pass filter (G)

In a signals analysis framework, the one-dimensional wavelet transform can be interpreted as the action of two filters on the original signal: the *high pass* filter and *low pass* filter, and (confusingly) these filters are given the conventional names G and H, respectively. As seen in Figure 3.5, the action of these filters is to separate the high and low frequency components of the signal, which divides the signal into the familiar approximation and difference coefficients of the WPT. [24] Note that the interpretation of the difference and approximation coefficients as the high and low frequency components of the signal corresponds, as we discussed earlier, to the interpretation of the approximation coefficients as the course-grained signal and the difference coefficients as representative of what must be added to the course-grained signal to return the original signal.

As seen in Figure 3.6, from [24], repeated passes on the approximation and difference coefficients with the high and low pass filters further separate out the various

frequency components of the signal, until each of the leaves in the wavelet packet tree cover a narrow frequency band (the $2\downarrow$ boxes denote the fact that the number of coefficients per leaf is reduced by a factor of two at every pass). But because of the way in which these frequency components are sorted out, the order of the packets in the full wavelet tree does not correspond to the order of increasing frequencies. As a result, the implementation of spectrum estimation requires us to establish the correspondence between *natural ordering*, that is, the natural order of the packets from left to right in Figure 3.6, and *frequency ordering*, which orders the components in terms of the frequency components of the signal they represent. At the bottom of Figure 3.6, the natural and frequency orderings for a one-dimensional WPT of depth 4 are given, but we need to know the general rule that establishes this correspondence.

This general rule is given by a *greycode* numbering scheme; further technical details about why this scheme applies can be found in [24]. A `n`-bit single-dimensional greycode numbering scheme `gc` can easily be generated by a recursive code, given by the following steps (in MATLAB code).

```
gc = [0, 1];
for j=2:n
    q = gc + 2^(j-1);
    q = fliplr(q);  % reverses the array column order
    gc = [gc, q];
end
```

Once this code has been established, we can easily associate the various leaves of the wavelet packet tree with the estimated frequencies. Because the WPT is bound by the Nyquist limit just like any other method of spectrum estimation, we know that the full range of frequencies that we are investigating must range from 0 to $.5F_s$, or half the sampling frequency. Given a wavelet packet transform of depth j,

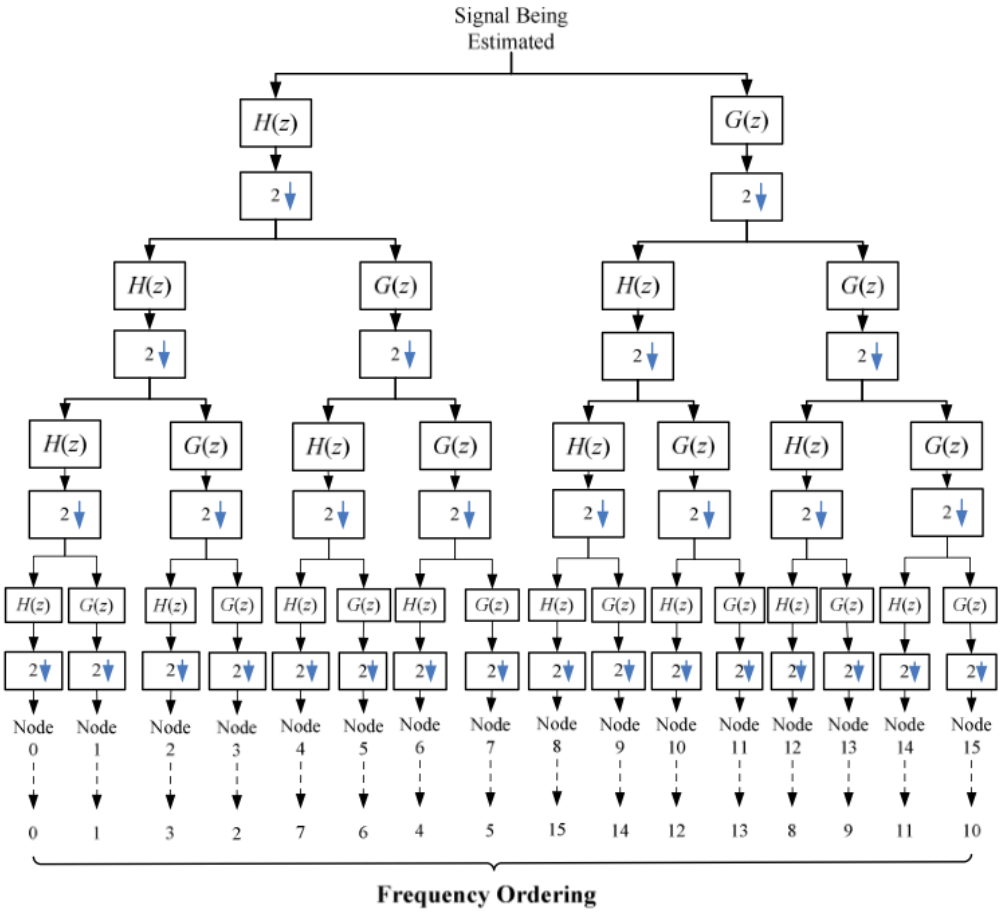

Figure 3.6: From [24]. A representation of a depth 4 WPT as the action of a low-pass filter (H) and a high-pass filter (G), with natural and frequency orderings denoted at the bottom.

which results in 2^j leaves at level j, we divide the frequency range $(0, .5F_s)$ into 2^j bins of equal length, and estimate the power spectrum as

$$P(f_i) = var[C_i] \tag{3.12}$$

where f_i is the ith frequency band, $var[C_i]$ represents the variance of the coefficients of the ith leaf *according to the frequency ordering.* [24]

Two-Dimensional Spectrum Estimation What we have described thus far is the justification for spectrum estimation in the case of the one-dimensional WPT. However, our research involves the two-dimensional WPT, so we will need to apply spectrum estimation techniques in this case as well. Fortunately, a similar process is involved.

The 2-dimensional WPT can be represented using the packet tree in Figure 2.10. As described in Chapter 2, each successive iteration or pass of the transform increases the number of packet leafs and decreases the number of coefficients per leaf by a factor of 4. Thus, for a given depth j, the number of leafs is 4^j. The natural ordering of these leafs can be indexed by $\{0, 1, ..., 4^j - 1\}$. For example, in Figure 2.10, at depth $j = 2$, there are 16 leafs, and the leaf indexing is given by reading off the leaves from left to right, so that we assign index 0 to leaf aa_{ij}, 1 to ad_{ij}, 7 to dh_{ij}, and so on.

Again, the Nyquist limit still applies to spectrum estimation with wavelets as it does with Fourier analysis, so we are still limited to examining the power at frequencies lower than half our sampling frequency. With this in mind, using the sampling frequencies F_x and F_y in the x and y directions, we compose a frequency grid with 2^j evenly spaced frequency bins between 0 and $.5F_x$ along the x-axis and between 0 and $.5F_y$ along the y-axis. This frequency grid is then a 2^j by 2^j matrix, as shown in Figure 3.7.

Instead of the simple greycode sequence used in the one-dimensional case, we generate a two-dimensional $2^j \times 2^j$ greycode matrix (the technical details of how this matrix is generated can be found in the code used in Appendix D). Each element of this greycode matrix then corresponds to a point on the frequency grid, as shown in Figure 3.7. We can define a function, g, such that $g(r, c)$ is equal to the number in the greycode matrix in row r and column c.

With these tools in place, we can then estimate the two-dimensional power spectrum

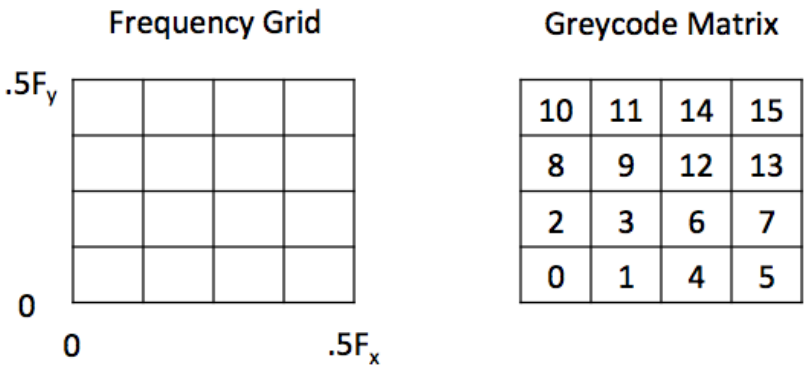

Figure 3.7: On the left, a frequency grid for a two-dimensional WPT of depth 2. On the right, the corresponding greycode matrix, which allows for the assignment of leaves of the wavelet packet tree to the individual bins in the frequency grid.

as

$$P(f_r, f_c) = var[C_{g(r,c)}] \tag{3.13}$$

where (f_r, f_c) corresponds to the bin in the frequency grid at row r and column c, and $var[C_{g(r,c)}]$ represents the variance of the coefficients in the leaf designated by $g(r, c)$ in natural ordering [25]. See Figure 3.8 for an example of the resulting two-dimensional power spectrum.

3.4 Radial Averaging

Having obtained our two-dimensional power spectrum $P_{2d}(k_x, k_y)$, we derive the one-dimensional power spectrum in the following way. Using the sampling frequencies in the x and y directions as guidelines, we establish k-bins corresponding to the points at which we want to estimate the power spectrum. Each cell (k_x, k_y) is assigned a overall frequency

$$k = \sqrt{k_x^2 + k_y^2} \tag{3.14}$$

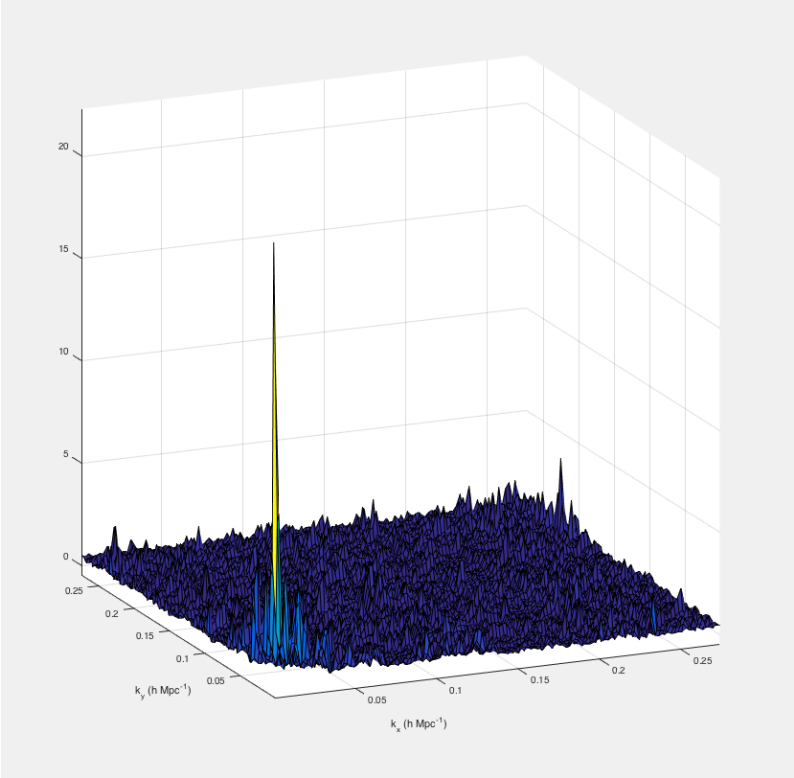

Figure 3.8: A sample two-dimensional power spectrum. Note that because we used SDSSPix, which has a different sampling rate across different axes, our k_x and k_y are not symmetric.

and placed in a k-bin accordingly. We then average over the power in each k-bin, such that if there are N data points in the k_ith bin

$$P_{1d}(k_i) = \frac{1}{N} \sum_{\sqrt{k_x^2+k_y^2} \in k_i} P(k_x, k_y) \tag{3.15}$$

The result is our radially averaged power spectrum, which we can then analyze to find the BAO signal. See Figure 3.9 for an example of the resulting radially averaged power spectrum.

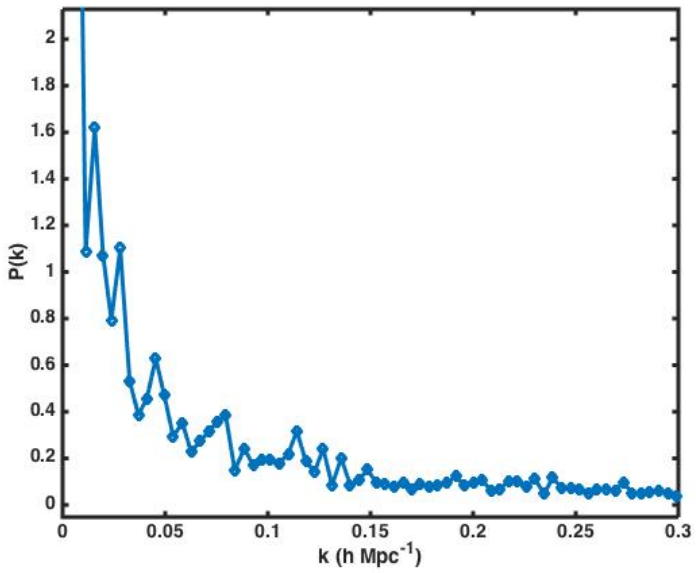

Figure 3.9: A sample radially averaged power spectrum.

CHAPTER 4

Data Analysis

In this chapter, we will analyze our data and present the results of our analysis. In Section 4.1, we will outline the methods of analysis that allow us to extract cosmological information from the one-dimensional power spectrum that we described in Chapter 3. In Section 4.2, we will outline the code that we developed and used in analyzing our data. In Section 4.3, we will present the actual results of this analysis using BOSS data.

4.1 Power Spectrum Analysis

As described in Chapter 1, we want to present our results in terms of the parameter α, which quantifies the deviation of our measured power spectrum from that of a fiduciary cosmological model. Thus, we will begin by describing our methods of comparing our results to the fiduciary cosmology. We will also introduce the parameter β, which is related to α, and describe how it is calculated.

4.1.1 Extracting the BAO Signal

Given an estimation of the power spectrum $P(k)$, we want to detect the BAO signature and compare this signature with a fiduciary cosmology. Since the BAO signal takes the form of an oscillatory component convolved with the background power spectrum, our priority is to extract this oscillatory function. We begin by finding a smooth cubic spline fit to the power spectrum, which represents the background

power spectrum and is labeled the 'smooth' power spectrum $P_{sm}(k)$. Since the oscillatory component is convolved with this function, we simply divide the original power spectrum by this smoothed power spectrum to get our oscillatory component,

$$O(k) = \frac{P(k)}{P_{sm}(k)} \tag{4.1}$$

Because our power spectrum is given in arbitrary units, this oscillation may not be easily comparable in terms of scale to other oscillations, which is what we ultimately want to do. To fix this problem, we 'normalize' this oscillatory component by a simple linear scaling, such that its amplitude is the same as the oscillation extracted from our fiducial cosmological model. This provides a useful basis for comparison, which we will now discuss.

4.1.2 Computing α

To compute α, we use the Cosmic Anisotropic Microwave Background software [16], to generate a fiduciary cosmology with parameters defined by BOSS [14]. Using the power spectrum of this fiduciary cosmology, we perform the same procedure outlined in Section 4.1.1 to extract the oscillatory component of the power spectrum, $O_{fid}(k)$

We then find the best fit the measured oscillatory function $O(k)$ to the fiduciary oscillation $O_{fid}(k)$ in terms of the parameter α

$$O(k) = O_{fid}(k/\alpha) \tag{4.2}$$

by finding the value of α which minimizes χ^2.

Functionally, α represents a measure of how much the fiduciary oscillation must be stretched or compressed in order to match the measured power spectrum. Figure 4.1 is an example of the two oscillations extracted from the measured power spectrum (red) and the fiduciary power spectrum (blue), respectively, with the fiduciary oscillation stretched by the parameter α as given in Equation 4.2 to give the best

fit. In particular, the best will be one in which the oscillatory behavior of the two functions match.

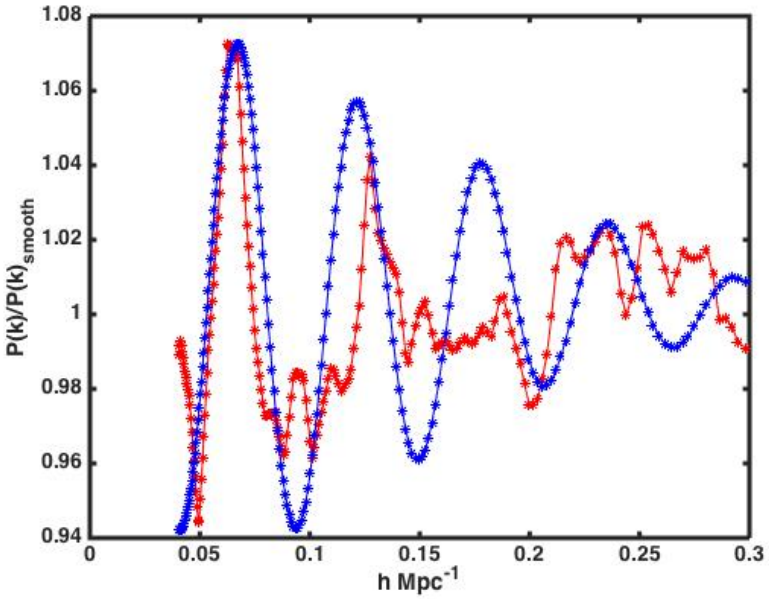

Figure 4.1: Comparison of the oscillatory components of the measured power spectrum (red) and the fiduciary power spectrum (blue), with the fiduciary power spectrum stretched by the α parameter to give the best fit.

4.1.3 Computing β

We compute β using the same process used to calculate α, with one key difference: instead of comparing our results directly to the fiduciary cosmology, we compare our results to the BOSS results recorded in [14]. In this case, because the oscillatory component of the power spectrum O_{BOSS} is given in the paper, there is no need to go through the process of computing the smooth power spectrum.

Further, we can relate our parameter β to α in the following way. The BOSS collaboration reported their results using the parameter α; however, since we want to distinguish the α that we report from their parameter, we will call their parameter

α_{BOSS} [14]

$$O_{BOSS}(k) = O_{fid}(k/\alpha_{BOSS}) \tag{4.3}$$

Thus, if we fit our oscillatory function to the BOSS data with the parameter β,

$$O(k) = O_{BOSS}(k/\beta) \tag{4.4}$$

Then a simple substitution of Equation 4.3 into Equation 4.4 gives us

$$O(k) = O_{fid}(k/(\beta\alpha_{BOSS})) \tag{4.5}$$

Thus, by reference to Equation 4.1, we can conclude that

$$\alpha = \beta\alpha_{BOSS} \tag{4.6}$$

Thus, we can use the parameter β as a measure of how well our data agrees with the BOSS results, and independently calculate α; in theory, these two numbers should be identical. As we will discuss in Section 4.3, however, this is not always the case.

4.2 Code

In this section, we present an overview of the code used in our analysis, as well as a description of the various subroutines that make it up. For a presentation of the actual code to the various subroutines described in this section, see Appendixes B-E. The various parts of the code were written collaboratively with Noel Garcia, Taurean Ford, Jesus Pando, and myself.

4.2.1 Overview and `main`

Figure 4.2, a reproduction of Figure 3.1, is an overview of the whole process. In this section, we will present the `main` routine and indicate various the subroutines which correspond to elements of Figure 4.2.

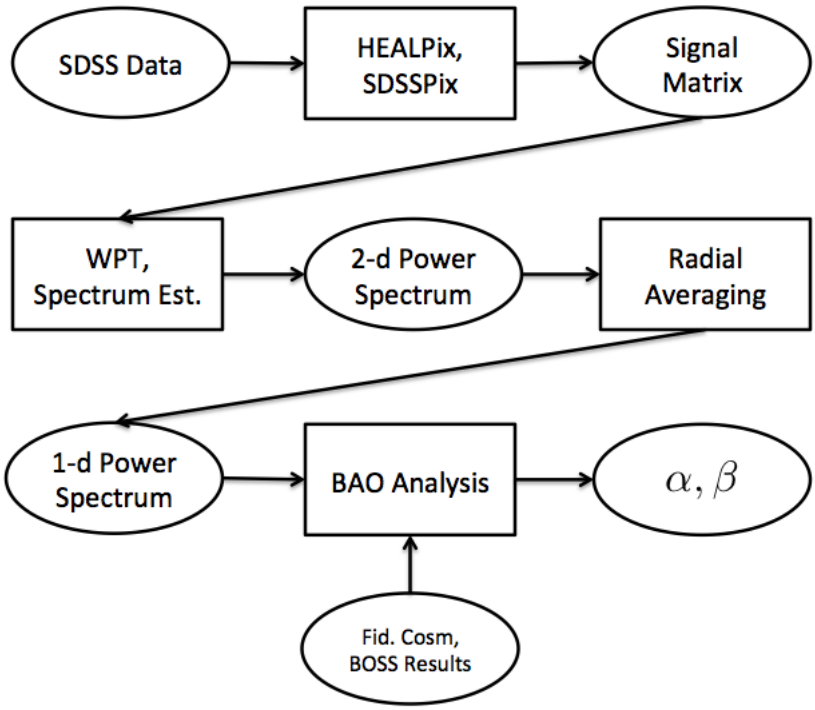

Figure 4.2: An overview flowchart of our entire process. Circles in the flowchart represent data inputs/outputs, while boxes represent the various processes and blocks of code that process data.

The main inputs to our program are `res`, which is the desired resolution for analysis; `ns`, which chooses if we want to examine data from the northern or southern galactic cap; `db`, which selects the specific wavelet that we want to use, e.g. `'db2'` for the Daubachies-2, etc.; `level`, for the desired depth of the WPT; and `cambdata`, which is the power spectrum of the fiducial cosmology that we compare our results to.

The two main outputs are `alpha` (α) and `beta` (β), the two parameters described in Section 4.1, though along the way we can generate graphs of the various power spectrums or other intermediary products if we so choose.

```
function [alpha,beta] = main(res, ns, db, level, cambdata)
    % Builds signal matrix
```

```
    pixels = sdsspix(res, ns);
    matrix = matrice(pixels, res, ns);
    reduced = chaffcut(matrix, res, ns);

    % Performs multilevel wavelet decomposition on our reduced 2^jmax by 2^jmax
    % matrix and spherically averages the 2d power spectrum.
    [k,Pk,Pksig] = WPS(reduced, res, level, db);

    % Finds alpha, beta
    [alpha] = alphafitter(cambdata,k,Pk,Pksig);
    [beta] = betafitter(k,Pk,Pksig);
end
```

4.2.2 Subroutines

In this section, we outline the subroutines that comprise our overall program. Note that some of the subroutines in this section are not found in the `main` routine outlined above, as they are subroutines of subroutines.

SDSSPix The first set of subroutines, corresponding to the top middle block in Figure 4.2, consists of `sdsspix`, `radec_to_lambdaeta`, `matrice`, and `chaffcut`. `sdsspix`, with the assistance of `radec_to_lambdaeta`, converts the raw RA-DEC coordinates provided in the SDSS datasets to SDSSPix indices for a given resolution using the theory described in Section 3.2.4 and outputs them as a vector array. We then use the `matrice` function to convert this array of SDSSPix indices into matrix form as described in Section 3.2.4. Finally, the program `chaffcut` cuts out a suitably sized 2^n square matrix for analysis.

WPT, Spectrum Estimation The subroutine `WPS` performs the wavelet packet transform on the signal matrix produced in the previous section, and outputs a one-dimensional power spectrum. Thus, this program corresponds to the entire middle row of Figure 4.2, performing both the WPT, the spectrum estimation, and the radial averaging process, and there are a number of subroutines that make this feasible.

The built-in MATLAB function `wpdec2` performs the two-dimensional wavelet packet transform on our signal matrix, and several other built-in functions from the MATLAB signal analysis toolbox, such as `wpcoef`, allow us flexibility in accessing the information in the wavelet packet tree. Our subroutine `est2dspec` generates the greycode matrix seen in Figure 3.7 and described in Section 3.3.2, and within the overall `WPS` we complete the process of estimating the two-dimensional power spectrum and average it into a one-dimensional power spectrum as described in Section 3.4.

Analysis Finally, given the power spectrum output by `WPS`, the programs `alphafitter` and `betafitter` solve for α and β as described in Section 4.1.

4.3 Results

The results presented in this section are the analysis results of the full program just outlined on the CMASS galaxy catalogues from BOSS Data Release 11. In particular, we run the data in both the northern and southern Galactic Caps at multiple resolutions, using two different kinds of edge extensions and six wavelet types.

As with any convolution on a finite length signal, border distortions arise. To deal with this issue, the boundaries of our signal need to be treated differently. Various

methods have been developed to handle these [26]. In this work we restricted our study to two of the simplest schemes edge extension methods, the *symmetric* and the *periodic* extension methods. The symmetric method assumes that the signal can be recovered outside the original range by a symmetric boundary value replication, while the periodic extension assumes that the signal outside the original range repeats periodically. Of course, in each case, there are drawbacks. In the former case, the symmetric extension creates artificial discontinuities in the first derivatives at the boundary, while in the latter case discontinuities in the original signal arise. However, any method of treating a finite signal will have these sorts of disadvantages, and as we discussed in Section 2.4 the localized nature wavelets keeps the error localized to just a few coefficients.

We confined our analysis to a total of six different Daubechies wavelets: the `db4`, `db8`, `db16`, `db22`, `db26`, and `db32` wavelets. These wavelets are similar to the Daubechies-2 wavelet described in Section 2.1.2, save that their recursive definitions involve more h_i coefficients that the h_0, h_1, h_2, and h_3 coefficients used in `db2`, and they are consequentially smoother. Fundamentally, however, the way that the wavelet transform works using this transform is not qualitatively different. We had no principled reason for choosing these particular Daubechies wavelets over others, save that this range of wavelet filters was expansive enough to find any significant effects due to wavelet choice. It is enough that these six allow us to check that the results we obtain among the various wavelet types are consistent.

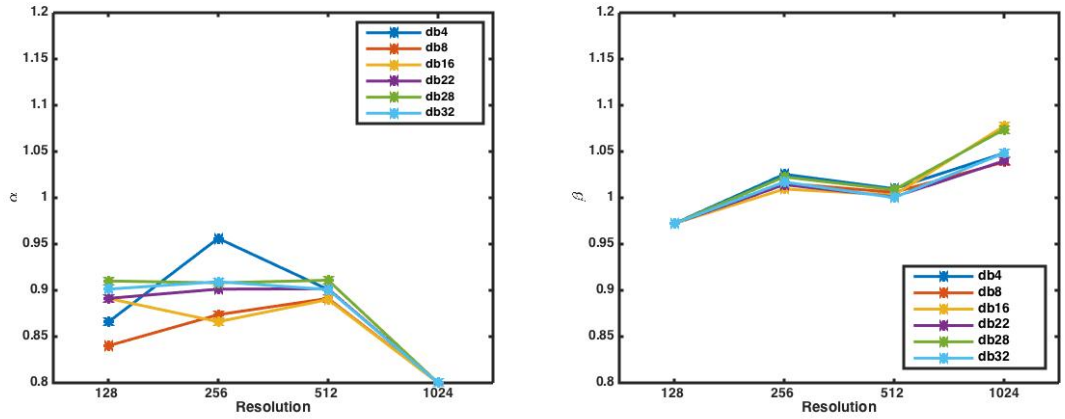

Figure 4.3: For the Northern dataset using periodic edge extensions. Left: (a) α. Right: (b) β.

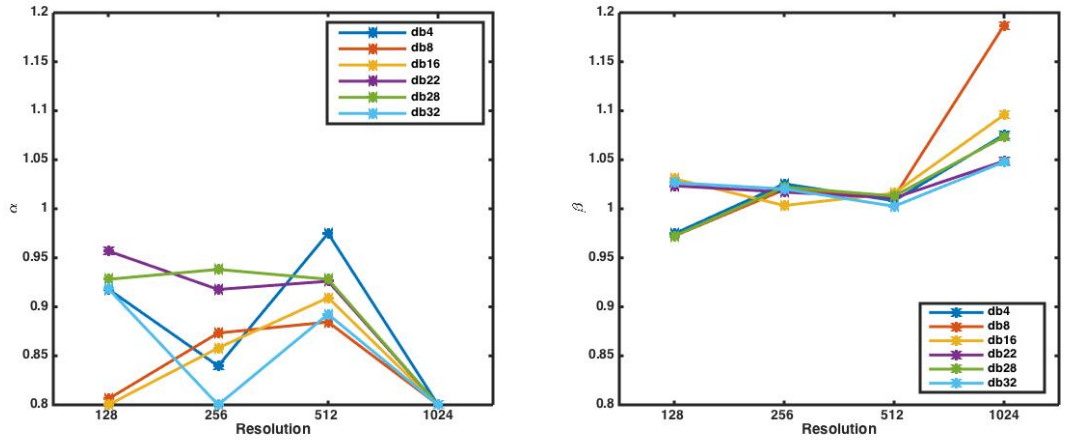

Figure 4.4: For the Southern dataset using periodic edge extensions. Left: (a) α. Right: (b) β.

Figures 4.3 and 4.4 contain the results from the northern and southern datasets, respectively, using periodic edge extensions (`'ppd'`). Figures 4.5 and 4.6 contain the results from the northern and southern data sets, respectively, using symmetric edge extensions (`'sym'`).

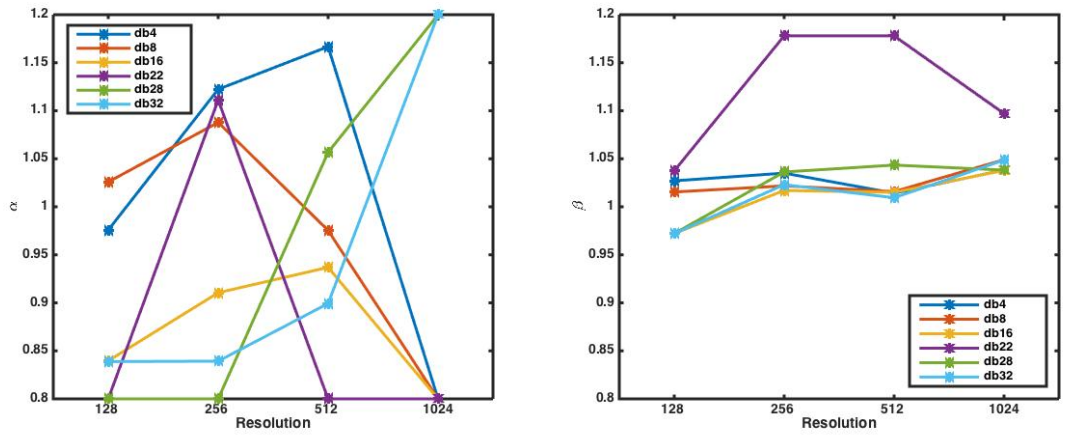

Figure 4.5: For the Northern dataset using symmetric edge extensions. Left: (a) α. Right: (b) β.

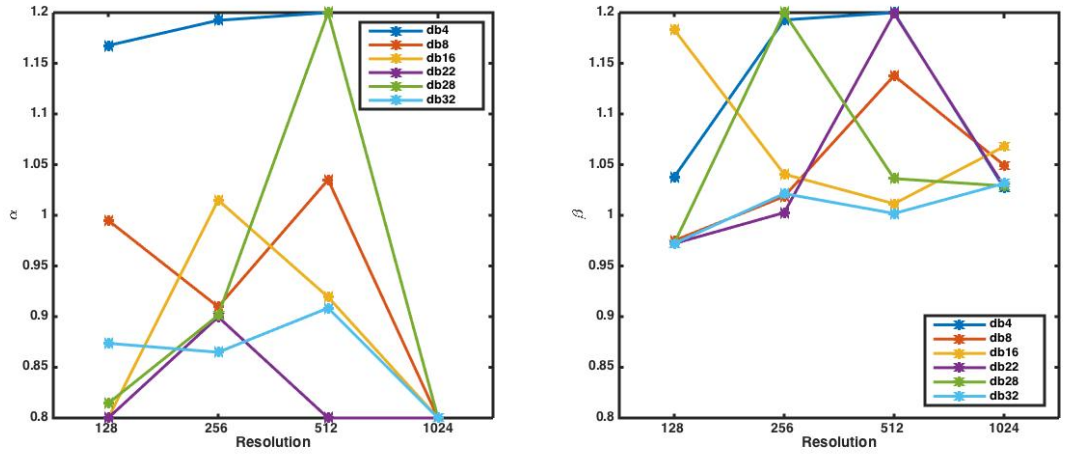

Figure 4.6: For the Southern dataset using symmetric edge extensions. Left: (a) α. Right: (b) β.

At each of the different resolutions, we operate at different physical scales. At resolution 128, the scale of an individual pixel is ≈ 1.8123 Mpc h^{-1}; at resolution 256, ≈ 0.9062 Mpc h^{-1}; at resolution 512, ≈ 0.4531 Mpc h^{-1}; and at resolution 1024, ≈ 0.2265 Mpc h^{-1}. The physical scale will affect our ability to capture the

BAO signal; for example, at larger physical scales the course-graining caused by the larger physical scale means that will miss small scale fluctuations.

Figure 4.7 gives the average values and spread ($\alpha_{max} - \alpha_{min}$) of α at the various resolutions and for both periodic and symmetric edge extensions, while Figure 4.8 does the same for β.

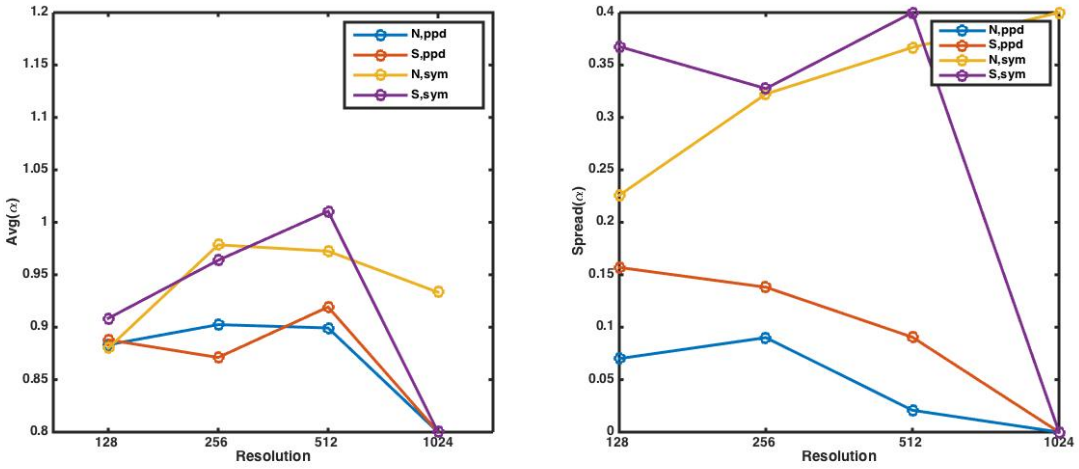

Figure 4.7: Left: (a) average value of α for various resolutions and edge extensions. Right: (b) the spread of α for the various resolutions and edge extensions.

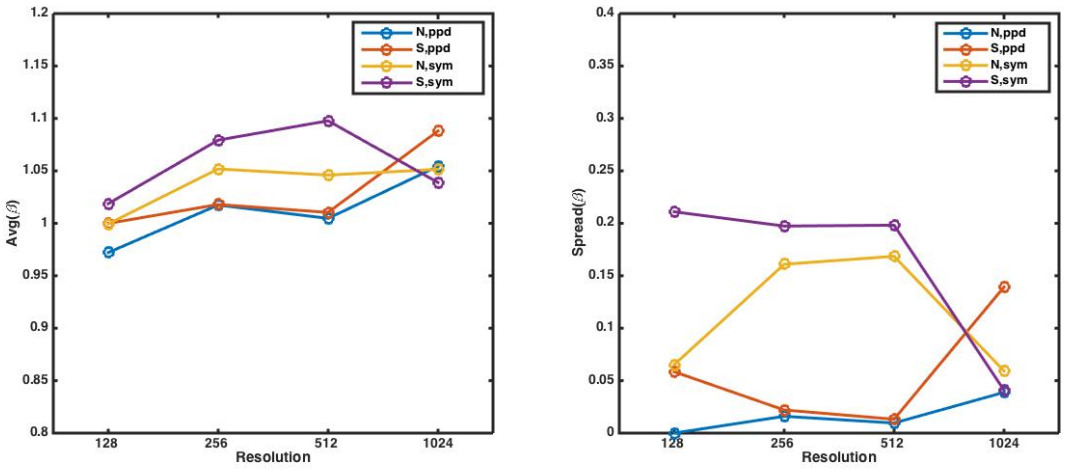

Figure 4.8: Left: (a) average value of β for various resolutions and edge extensions. Right: (b) the spread of β for the various resolutions and edge extensions.

As can be seen, overall our results for β are $\beta \approx 1$, indicating good agreement with the SDSS collaboration results. The result for α show much more variance. The SDSS collaboration obtained a value of $\alpha = 1.033$ and our results range from $\alpha = 0.8$ to $\alpha = 1.2$. The results using the periodic edge extensions give us much more consistent results in both α and β than the symmetric edge extensions, as can be seen from the fact that the spreads of both α and β are much lower in the former case (Spread(α) $\approx$ .1, Spread(β) $\approx$ .05) than in the latter case (Spread(α) $\approx$ .3, Spread(β) $\approx$= .15). The northern dataset also tends to give much more consistent results than the southern dataset—for almost every resolution and type of edge extension, if we compare Spread(α) and Spread(β) from the northern dataset to the southern dataset, then the former will be smaller. In the following chapter, we will interpret these results and discuss their implications.

CHAPTER 5

Conclusions and Future Research

In this section, we will analyze the the results of our research and discuss their implications for future research. In Section 5.1, we will analyze the results presented in Section 4.3, paying particular attention to which aspects of our project were successful and which deserve further inquiry. In section 5.2, we will discuss how these interpretations bear on the possibility of future research.

5.1 Interpretation of Results

The parameters α and β measure, respectively, how well our results match the fiduciary cosmology and the SDSS collaboration results. We interpret our results for each of these parameters individually.

First of all, our results for the value of the β parameter are excellent in most cases. All of our wavelet analysis produces values of $\beta \approx 1$. Because β is a parameter that measures how well we have managed to replicate the results of the SDSS's BAO analysis, we can be confident that our methods of spectrum estimation have successfully managed to detect the BAO signal according to our aims. Moreover, we have detected the BAO without the need for any free parameters or secondary manipulations. Furthermore, the different wavelets used in the transform—`db2`, `db16`, etc.— all produce consistent results. The exception to this success is the southern dataset under symmetric edge extensions, which is likely because the southern dataset is smaller.

Our calculated values for α have not been quite as consistent with BOSS results, where $\alpha \approx 1.033$. Indeed, some of our calculated values for α are so low that they likely fall outside of the range in which we searched for α, $[.8, 1.2]$, and it is likely that if we widened the range of our search our results would be even lower.

The difference in reproducing the SDSS's results as measured by β while failing to do so for α requires explanation. Our code produces a single power spectrum, and this power spectrum is used to calculate both α and β. Thus, the fact that the former results are deficient while the latter results are excellent must be due to the differences in the manner in which these parameters are calculated. But if this is the case, then the problem in our analysis is downstream of the actual power spectrum production, and our results in calculating β still represent a success.

In the case of the `alphafitter` program, we acquire the oscillatory component O_{fid} by extracting it from the full fiduciary power spectrum by dividing out the smooth power spectrum. By design, our method of extracting the BAO signal from the fiduciary power spectrum was much simpler than the method used by SDSS. As we discussed in Section 1.3.2, the SDSS fitting method shown in Equation 1.13 involves more than a few 'nuisance parameters' which were designed to produce an optimal fit. However, this also introduces some arbitrariness into the calculation of the BAO signal, which is something that we intentionally tried to avoid. Thus, while our method has not been quite as successful at reproducing α from the fiduciary power spectrum, our method is much simpler and there is less of a risk that we will introduce bias into our results.

Furthermore, when we run the SDSS power spectrum through the same `alphafitter` program that we use to analyze our own power spectrum, we get a value of $\alpha = .9838$, inconsistent with the expected result of $\alpha \approx 1.033$. Thus, our methodology in `alphafitter` also fails to reproduce SDSS results for α even when we analyze SDSS's own power spectrum. The fact that our simpler method

of calculating α also produces a much lower result than expected when applied to the SDSS power spectrum is also a strong indication that the issue lies with that particular method, and not our power spectrum generation procedures.

On the other hand, in our `betafitter` program, this method of extraction is unnecessary because we can use the values for O_{BOSS} directly from the SDSS paper [14]—in effect, the oscillatory component that we use to compare our results comes pre-extracted, and our results for β reflect this. Thus, we have indeed managed to extract the same information as the SDSS collaboration with wavelets, which was our primary objective. Using the relationship between α and β outlined in Section 4.1.3 and described in Equation 4.6, we can estimate our expected value for α had we processed our power spectrum using the SDSS's full methodology. Depending on the resolution and edge extension method used, we get values ranging from $\alpha = 1.002$ to $\alpha = 1.136$, which tend to be much closer to the desired results.

Another possible source of error is in the generation of our fiduciary cosmology. When we use CAMB [16] to generate our fiduciary cosmology, this gives us the matter power spectrum immediately after recombination *before* any nonlinear structure formation. And while it is true that the regime we are examining ($z \approx .5$) the effects of nonlinear structure formation are generally negligible, it is still a possible source of error.

The fact that our values for α are always outside of this $[.8, 1.2]$ range when we measure the signal at resolution 1024 is likely the result of Poisson noise. At these high resolutions, when the area of a given pixel is extremely tiny, our matrix signal becomes so sparse that shot noise drowns out our signal.

Although it is a good sign that our values for α are relatively consistent under periodic edge extensions, the wide range of values under symmetric edge extensions likely indicates that some elements of that method are throwing off our analysis.

Finally, our results from northern datasets tend to be better than the results from southern datasets. This is unsurprising, as the northern dataset is much larger than the southern dataset, and more data generally means that we will get better results. Indeed, the fact that our results become better when we use the better dataset is an indication that our methods are working—if we got equally good results regardless of the quality of our dataset, we might be suspicious that some element of our method was causing us to get good results that were unwarranted by the data.

5.2 Future Research

As we discussed in Section 1.3.3, the research in this thesis aimed to develop a method that could, as a litmus test, successfully reproduce the results from the SDSS. And as we have just established, we have been successful in that regard, although our hope of using a simpler smoothing function is only partially successful. Thus, the first clear avenue of future research is to apply these algorithms to the newest datasets from the SDSS—including Data Release 13, which is to be released at the end of July 2016.

On the subject of our partial success, there remain ways in which further research could improve on our methods. For instance, one might find a better manner of extracting the oscillatory component of the power spectrum, which would improve the results for our calculations of α. It is possible, however, that the entire method used in our `alphafitter` program could be improved. That is, perhaps instead of isolating the oscillatory component of the power spectrum for analysis, we could instead try to fit the entire power spectrum in the manner that SDSS does (as described in Section 1.3.2).

Other methods of analysis may be possible as well. For example, as part of our research, we attempted to find an effective method of taking the Hankel transform

of our power spectrum to obtain the two-point correlation function, as this would allow for a very simple analysis of the BAO signal. Unfortunately, over the course of our research, we found serious numerical problems with taking the Hankel transform. However, it might be that further research would reveal some avenues that we overlooked.

Finally, we have shown that wavelet packet methods can be effective tools in this sort of astrophysical research. Further research into the applications of wavelets may thus yield other interesting ways in which they can be used to advance astrophysical research. For example, our next avenue of research is attempting to use wavelet techniques to make calculation of the two-point and three point correlation functions less computationally expensive. As a corollary, wavelet methods may be useful in calculating higher-order spectra such as the bispectrum which could help us to characterize the BAO signal in other respects.

Appendices

APPENDIX A

Proving Parseval's Theorem for Wavelet Transforms

Our goal is to demonstrate

$$\int |\tilde{f}_i(x)|^2 \, dx = \int |\tilde{f}_f(x)|^2 \, dx \tag{A.1}$$

where $f_i(x)$ and $f_f(x)$ are the forms of the signal before and after undergoing a single pass of the wavelet transform, respectively,

$$\sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x) \xrightarrow{WPT} \sum_{j=0}^{2^{N-1}-1} a_j \phi_{[x_{2j}, x_{2(j+1)})}(x) + \sum_{j=0}^{2^{N-1}-1} d_j \psi_{[x_{2j}, x_{2(j+1)})}(x)$$

Note that because the operation of a single pass is the same for both the DWT and WPT, we do not need to distinguish between the two. We begin with the LHS of the equation,

$$\int |\tilde{f}_i(x)|^2 \, dx = \int \left| \sum_{j=0}^{2^N-1} s_j \phi_{[x_j, x_{j+1})}(x) \right|^2 dx = \sum_{j=0}^{2^N-1} \sum_{k=0}^{2^N-1} \int s_j s_k \phi_{[x_j, x_{j+1})}(x) \phi_{[x_k, x_{k+1})}(x) \, dx$$

By exploiting the orthogonality of our scaling functions, namely that

$$\int \phi_{[x_j, x_{j+1})}(x) \phi_{[x_k, x_{k+1})}(x) = \delta_{jk}[x_{j+1} - x_j]$$

we can conclude that

$$\int |\tilde{f}_i(x)|^2 \, dx = \sum_{j=0}^{2^N-1} \sum_{k=0}^{2^N-1} s_j s_k \delta_{jk}[x_{j+1} - x_j]$$

and exploiting the delta function and the fact that $x_{j-1} - x_j = 1/f_s$, where f_s is the sampling frequency, for all j,

$$\int |\tilde{f}_i(x)|^2 \, dx = \frac{1}{f_s} \sum_{j=0}^{2^N-1} |s_j|^2$$

Next, we examine the RHS of the equation,

$$\int |\tilde{f}_f(x)|^2 \, dx = \int \left| \sum_{j=0}^{2^{N-1}-1} a_j \phi_{[x_{2j}, x_{2(j+1)})}(x) + \sum_{j=0}^{2^{N-1}-1} d_j \psi_{[x_{2j}, x_{2(j+1)})}(x) \right|^2 dx$$

Though this is slightly more complicated, this is fundamentally the same as the LHS. We simply expand, with some clever manipulation of indices,

$$\int |\tilde{f}_f(x)|^2 \, dx = \int \sum_{j=0}^{2^{N-1}-1} \sum_{k=0}^{2^{N-1}-1} a_j a_k \phi_{[x_{2j}, x_{2(j+1)})}(x) \phi_{[x_{2k}, x_{2(k+1)})}(x) +$$

$$2 \sum_{j=0}^{2^{N-1}-1} \sum_{k=0}^{2^{N-1}-1} a_j d_k \phi_{[x_{2j}, x_{2(j+1)})}(x) \psi_{[x_{2k}, x_{2(k+1)})}(x) + \sum_{j=0}^{2^{N-1}-1} \sum_{k=0}^{2^{N-1}-1} d_j d_k \psi_{[x_{2j}, x_{2(j+1)})}(x) \psi_{[x_{2k}, x_{2(k+1)})}(x)$$

Again, we exploit the orthogonality of wavelets; however, this time we need three separate relations,

$$\int \phi_{[x_{2j}, x_{2j+2})}(x) \phi_{[x_{2k}, x_{2k+2})}(x) = \delta_{jk}[x_{2j+2} - x_{2j}] = \delta_{jk} \frac{2}{f_s}$$

$$\int \psi_{[x_{2j}, x_{2j+2})}(x) \psi_{[x_{2k}, x_{2k+2})}(x) = \delta_{jk}[x_{2j+2} - x_{2j}] = \delta_{jk} \frac{2}{f_s}$$

$$\int \phi_{[x_{2j}, x_{2j+2})}(x) \psi_{[x_{2k}, x_{2k+2})}(x) = 0$$

The center term will drop out, while the remaining two terms leave behind factors,

$$\int |\tilde{f}_f(x)|^2 \, dx = \sum_{j=0}^{2^{N-1}-1} \sum_{k=0}^{2^{N-1}-1} a_j a_k \delta_{jk} \frac{2}{f_s} + \sum_{j=0}^{2^{N-1}-1} \sum_{k=0}^{2^{N-1}-1} d_j d_k \delta_{jk} \frac{2}{f_s}$$

Applying delta functions,

$$\int |\tilde{f}_f(x)|^2 \, dx = \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} |a_j|^2 + \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} |d_j|^2$$

The equivalence is easiest to show for the Haar wavelet, but it can be shown in principle be shown for any wavelet. By substituting in for a_j and d_j using the relationships in Equations 2.20 and 2.21,

$$\int |\tilde{f}_f(x)|^2 \, dx = \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} \left| \frac{s_{2j} + s_{2j+2}}{2} \right|^2 + \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} \left| \frac{s_{2j} - s_{2j+2}}{2} \right|^2$$

$$\int |\tilde{f}_f(x)|^2\, dx = \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} \frac{|s_{2j}|^2 + 2s_{2j}s_{2j+2} + |s_{2j+2}|^2}{4} + \frac{2}{f_s} \sum_{j=0}^{2^{N-1}-1} \frac{|s_{2j}|^2 - 2s_{2j}s_{2j+2} + |s_{2j+2}|^2}{4}$$

By canceling the extra factors of 2 and the terms of $2s_{2j}s_{2j+2}$,

$$\int |\tilde{f}_f(x)|^2\, dx = \frac{1}{f_s} \sum_{j=0}^{2^{N-1}-1} \frac{|s_{2j}|^2 + |s_{2j+2}|^2}{2} + \frac{1}{f_s} \sum_{j=0}^{2^{N-1}-1} \frac{|s_{2j}|^2 + |s_{2j+2}|^2}{2}$$

These are the same; thus,

$$\int |\tilde{f}_f(x)|^2\, dx = \frac{1}{f_s} \sum_{j=0}^{2^{N-1}-1} |s_{2j}|^2 + |s_{2j+2}|^2$$

But since this represents two summations—one of all the even terms between $j = 0$ and $2^N - 1$, and the other the odd terms between the same limits, we can rewrite this as a single summation,

$$\int |\tilde{f}_f(x)|^2\, dx = \frac{1}{f_s} \sum_{j=0}^{2^N-1} |s_j|^2$$

and this is precisely equivalent to what we showed the RHS of the equation was. Thus,

$$\int |\tilde{f}_i(x)|^2\, dx = \int |\tilde{f}_f(x)|^2\, dx$$

APPENDIX B

sdsspix Program

```
function [hindex] = sdsspix(res, ns)
    % Setting base pixel resolution, according to SDSS standards
    nx0 = 36;
    ny0 = 13;

    % Setting other important constants
    d2r = pi/180;
    r2d = 180/pi;
    etaOffSet = 91.25;
    surveyCenterRA = 185.0;
    surveyCenterDEC = 32.5;
    node = d2r * (surveyCenterRA - 90.0);
    etaPole = d2r * surveyCenterDEC;

    % Loading RA-DEC Coordinates
    if ns == 1
        info = fitsinfo('/Users/.../galaxy_DR11v1_CMASS_North.fits');
        data = fitsread('/Users/.../galaxy_DR11v1_CMASS_North.fits',...
        'binarytable','info',info,'TableColumns',[1 2]);
    else
        info = fitsinfo('/Users/.../galaxy_DR11v1_CMASS_South.fits');
        data = fitsread('/Users/.../galaxy_DR11v1_CMASS_South.fits',...
```

```
    'binarytable','info',info,'TableColumns',[1 2]);
end
ra = data{1,1};
dec = data{1,2};

% Converting RA-DEC -> LAMBDA-ETA
[lambda,eta] = radec_to_lambdaeta_SDSSPIX(ra,dec);

% Converting LAMBDA-ETA -> HEALPix index
nx = nx0*res;
ny = ny0*res;

% In parallel, calculate the healpix index for each lambda-eta coord.
parfor n = 1:length(eta)
    eta1=(eta(n)-etaOffSet)*d2r;
    if eta1 >= 0
        eta2 = eta1;
    else
        eta2 = eta1 + 2*pi;
    end
    i = floor((nx*eta2)/(2*pi));

    lambda2 = (90 - lambda(n))*d2r;
    if lambda2 >= pi
        j = ny - 1;
    else
        j = floor(ny*((1-cos(lambda2))/2));
    end
```

```
            hindex(n) = nx*j+i;
        end

    end
```

APPENDIX C

matrice Program

```
function [matrix] = matrice(pixels, res, ns)
    % Setting parameters
    nx0 = 36;
    ny0 = 13;
    nx = nx0 * res;
    ny = ny0 * res;

    % Loading data on weights
    if ns == 1
        info = fitsinfo('/Users/.../galaxy_DR11v1_CMASS_North.fits');
        data = fitsread('/Users/.../galaxy_DR11v1_CMASS_North.fits',...
        'binarytable','info',info,'TableColumns',[26]);
    else
        info = fitsinfo('/Users/.../galaxy_DR11v1_CMASS_South.fits');
        data = fitsread('/Users/.../galaxy_DR11v1_CMASS_South.fits',...
        'binarytable','info',info,'TableColumns',[26]);
    end

    weight = data1,1;

    matrix = zeros(nx, ny); % Pre-allocating
```

```
    % Constructing matrix
    for n = 1:length(pixels);
        column = floor(pixels(n)/nx) + 1;
        row = mod(pixels(n), nx) + 5 * res;

        if row > nx
            row = row - nx;
        end

        matrix(row, column) = matrix(row, column)+weight(n);
        clear row;
        clear column;
    end

end
```

APPENDIX D

WPS Program

```
function [k,Pk,Pksig] = WPS(pixels, res, ns)
    % Setting DWT mode, finding Fs
    dwtmode('ppd', 'nodisp');
    [~,Fs] = res_to_freq(res);
    [Fsx,Fsy] = res_to_freq2d(res);

    % Generating grid and corresponding 2d-k-freq bins
    [grid] = est2dspec(j);
    grid = grid+1;

    % Generating the tree for the 2-d wavelet packet transform
    T = wpdec2(x, j, filter);

    % Estimating 2-d spectrum
    Ps2d = grid;
    for p = 1:4^j
        [xi,yi] = find(grid == p);
        coefs = wpcoef(T,[j, p-1]);
        Ps2d(xi,yi) = var(coefs(:));
    end

    %% Spherically averaging our 2d power spectrum into linear k-bins
```

```
pbounds = 0:1:(2^j);
avgbands = (0:1:(floor((2^j)*sqrt(2))))*.5*Fs/(2^j);
xbands = pbounds*.5*Fsx/(2^j);
ybands = pbounds*.5*Fsy/(2^j);

% Setting up equally-spaced frequency bins between [0,.5*Fs].
kxbins = zeros(1,length(xbands)-1);
kybins = zeros(1,length(xbands)-1);
kbins = zeros(1,length(avgbands)-1);
parfor i = 1:length(xbands)-1
    kxbins(i) = (xbands(i)+xbands(i+1))/2;
    kybins(i) = (ybands(i)+ybands(i+1))/2;
    kbins(i) = (avgbands(i)+avgbands(i+1))/2;
end

% Converting to polar coordinates
[fv1,fv2] = meshgrid(kxbins,kybins);
[~, rho] = cart2pol(fv1, fv2);

% creating a matrix, prho, which assigns each leaf to a kbin index
prho = rho;
for i = 1:length(rho)
    for j = 1:length(rho)
        [~,k] = min(abs(kbins-rho(i,j)));
        prho(i,j) = k-1;
    end
end
```

```
    % Radially averaging
    sz = length(Ps2d);
    i = cell(sz+1,1);

    for r = 0:sz
        ir+1 = find(prho==r);
        Pk(r+1) = nanmean(Ps2d(ir+1));
        Pksig(r+1) = nanstd(Ps2d(ir+1));
        k(r+1) = nanmean(rho(ir+1));
    end


end
```

APPENDIX E

alphafitter Program

```
function [alpha] = alphafitter(cambdata, k, Pk, Pksig)
    % load fiduciary data
    k_fid = cambdata(:,1);
    Pk_fid = cambdata(:,2);

    % split Pk_fid into Pksm_fid and O_fid
    logPksm_fid = csaps(log10(k_fid),log10(Pk_fid),.9);
    Pksm_fid = 10.^fnval(logPksm_fid,log10(k_fid));
    O_fid = Pk_fid./Pksm_fid;

    % Smooth out Pk into Pksm and O
    logPksm = csaps(log10(k),log10(Pk),.9);
    Pksm = 10.^fnval(logPksm,log10(k));
    O = Pk./Pksm;

    % Scale O to match amplitude of O_fid
    [~,imin] = min(abs(k-.03));
    [~,imax] = min(abs(k-.3));
    ymin = min(O_fid);
    ymax = max(O_fid);
    xmin = min(O(imin:imax));
    xmax = max(O(imin:imax));
```

```
a = (ymax-ymin)/(xmax-xmin);
b = ymax - a*xmax;
O = a.*O+b;

% Fit data
deltaalpha = 1000;
rng = .2;
alpharange = linspace(1-rng,1+rng,deltaalpha);

kmin = max([.04,k_fid(1),k(1)]);
kmax = min([.3,k_fid(end),k(end)]);

a = find(k_fid > kmax);
k_fid(a) = [];

b = find(k_fid < kmin);
k_fid(b) = [];

for i = 1:deltaalpha
    O_stretch = interp1(k_fid,O_fid,k_fid/alpharange(i));
    O2 = interp1(k,O,k_lin2);
    chi2 = ((O_stretch-O2).^2);
    sumchi2(i) = sum(chi2);
end

[~,j] = min(sumchi2);
alpha = alpharange(j);
```

```
end
```

REFERENCES

[1] L. Anderson et al. The clustering of galaxies in the SDSS-III Baryon Oscillation Spectroscopic Survey: Baryon Acoustic Oscillations in the Data Release 10 and 11 galaxy samples. *MNRAS* 441 (1): 24-62, 2014. doi:10.1093/mnras/stu523.

[2] D.J. Eisenstein, H. Seo, & M. White. On the Robustness of the Acoustic Scale in the Low-Redshift Clustering of Matter. *ApJ* 664: 660-674, 2007. doi:10.1086/518755. arXiv:astro-ph/0604361.

[3] B.A. Bassett & R. Hlozek. Baryon Acoustic Oscillations. arXiv:0910.5224.

[4] D.J. Eisenstein. Baryon Acoustic Oscillations: A Robust and Precise Route to the Cosmological Distance Scale (Lecture). Harvard, 2009.

[5] E. Aubourg et al. Cosmological implications of baryon acoustic oscillation (BAO) measurements. *Phys. Rev. D* 92: 123516, 2015. doi: 10.1103/PhysRevD.92.123516.

[6] P.J.E. Peebles. *Large-Scale Structure of the Universe*. Princeton: Princeton, 1980.

[7] A.J.S. Hamilton. Power Spectrum Estimation I. Basics. arXiv:astro-ph/0503603

[8] W.J. Percival. Large Scale Structure Observations. arXiv:1312.5490.

[9] Alam et al. The Eleventh and Twelfth Data Releases of the Sloan Digital Sky Survey: Final Data from SDSS-III *ApJS* 219 (1): 12, 2015. doi:10.1088/0067-0049/219/1/12. arXiv:1501.00963v3.

[10] A.S. Bolton et al. Spectral Classification and Redshift Measurement for the SDSS-III Baryon Oscillation Spectroscopic Survey. *Astron.J.*, 144: 144, 2012. doi:10.1088/0004-6256/144/5/144. arXiv:1207.7326v2.

[11] F. Beutler et al. The clustering of galaxies in the SDSS-III Baryon Oscillation Spectroscopic Survey: Signs of neutrino mass in current cosmological datasets. *MNRAS* 444 (4): 3501-3516. doi: 10.1093/mnras/stu1702. arXiv:1403.4599v2.

[12] A.J. Cuesta et al. The clustering of galaxies in the SDSS-III Baryon Oscillation Spectroscopic Survey: Baryon Acoustic Oscillations in the correlation function of LOWZ and CMASS galaxies in Data Release 12. *MNRAS* 457 (2):1770-1785, 2016. doi: 10.1093/mnras/stw066. arXiv:1509.06371.

[13] S.D. Landy and A.S. Szalay. Bias and Variance of Angular Correlation Functions. *ApJ*, 412 (1): 64-71, 1993. doi:10.1086/172900.

[14] L. Anderson et al. The clustering of galaxies in the SDSS-III Baryon Oscillation Spectroscopic Survey: baryon acoustic oscillations in the Data Release 9 spectroscopic galaxy sample. *MNRAS* 427 (4): 3435-3467, 2012. doi: 10.1111/j.1365-2966.2012.22066.x. arXiv:1203.6594.

[15] H.A. Feldman, N. Kaiser, & J.A. Peacock. Power-spectrum analysis of three-dimensional redshift surveys. *ApJ* 426 (1): 23-37, 1994. doi:10.1086/174036.

[16] A. Lewis, A. Challinor, & A. Lasenby. Efficient computation of CMB anisotropies in closed FRW models. *ApJ* 538: 473-476, 2000. doi:10.1086/309179. arXiv:astro-ph/9911177.

[17] Y. Nievergelt. *Wavelets Made Easy*. Boster: Birkh auser, 1999.

[18] I. Daubechies. *Ten Lectures on Wavelets*. Philadelphia: SIAM, 1992.

[19] A.J. Ross et al. The clustering of galaxies in the SDSS-III Baryon Oscillation Spectroscopic Survey: analysis of potential systematics. *MNRAS* 424 (1): 564-590, 2012. doi:10.1111/j.1365-2966.2012.21236.x. arXiv:1203.6499v3.

[20] A.J. Ross et al. Ameliorating systematic uncertainties in the angular clustering of galaxies: a study using the SDSS-III. *MNRAS* 417 (2): 1350-1373, 2011. doi:10.1111/j.1365-2966.2011.19351.x.

[21] S. Ho et al. Clustering of Sloan Digital Sky Survey III Photometric Luminous Galaxies: The Measurement, Systematics, and Cosmological Implications. *ApJ* 761 (1): 14, 2012. doi:10.1088/0004-637X/761/1/14. arXiv:1201.2137.

[22] K.M. Gorski et al. The HEALPix Primer. arXiv:astro-ph/9905275v2.

[23] L. Fang & J. Pando. Large-scale Structures revealed by Wavelet Decomposition. arXiv:astro-ph/9701228.

[24] D.D. Ariananda. *On Wavelet Based Spectrum Estimation for Dynamic Spectrum Access* (PhD thesis). The Netherlands: Delft University of Technology, 2009.

[25] A.M. Atto, Y. Berthoumieu, & P. Bolon. 2-Dimensional Wavelet Packet Spectrum for Texture Analysis. *IEEE Transactions on Image Processing* 22 (6): 2013. doi:10.1109/TIP.2013.2246522.

[26] A. Cohen, I. Daubechie, and J.C. Feauveau. Biorthogonal basis of compactly supported wavelets. *Comm. Pure. Appli. Math* 45: 485-560, 1992. doi:10.1002/cpa.3160450502